Heat transfer investigations in a modern diesel engine

Finol Parra, Carlos

Award date:
2008

Awarding institution:
University of Bath

Appendix A. Instruments and Materials Specifications

A.1 K-Type Thermocouples

Mineral insulated thermocouple, nickel chromiun/nickel aluminium, plain stainless steel pot and stainless steel sheath, with 2 mm insulated lead and 7/0.2 mm flat pair.

Probe diameter: 1 mm, length: 250 mm.

Temperature range: -100 to 1100°C.

Accuracy: ±0.5ºC.

A.2 Fast Response Thermocouples

Nanmac pencil probe "eroding" thermocouples, one thermally grounded (E12-2-K) to measure the surface metal temperature and one thermally ungrounded (E12-2-K-U) to measure the interface or gas temperature at the surface.

Stainless steel body, probe diameter: 0.1875 mm, length: 152 mm.

Standard K-type thermocouple calibration, temperature range: 0-1300°C.

Pressure range: up to 685 bar.

Response time: low millisecond range.

A.3 Platinum Resistance Thermometers (PRT)

Hose PRTs, stainless steel body, diameter: 3 mm, lengths: 21, 11.5 and 15 mm.

PRTs calibrated at 80, 90, 100 and 110 ºC.

Maximum operating temperature: 230 ºC.

Accuracy: ±0.5ºC.

A.4 Flow Meters

Turbine flow meters Platon type FT13, FT22 and FT23.

Flow ranges: 10-80 l/min (FT13), 20-140 l/min (FT22), 10-160 l/min (FT13).

A.5 Data Loggers

Stand alone general purpose low power DataTakers Series 600 for data acquisition in real time. The accompanying software DeLogger allows remote monitoring, control and data management.

Channels: 10 to 30 sensor channels and 7 digital channels.

Data points: up to 1,390,000.

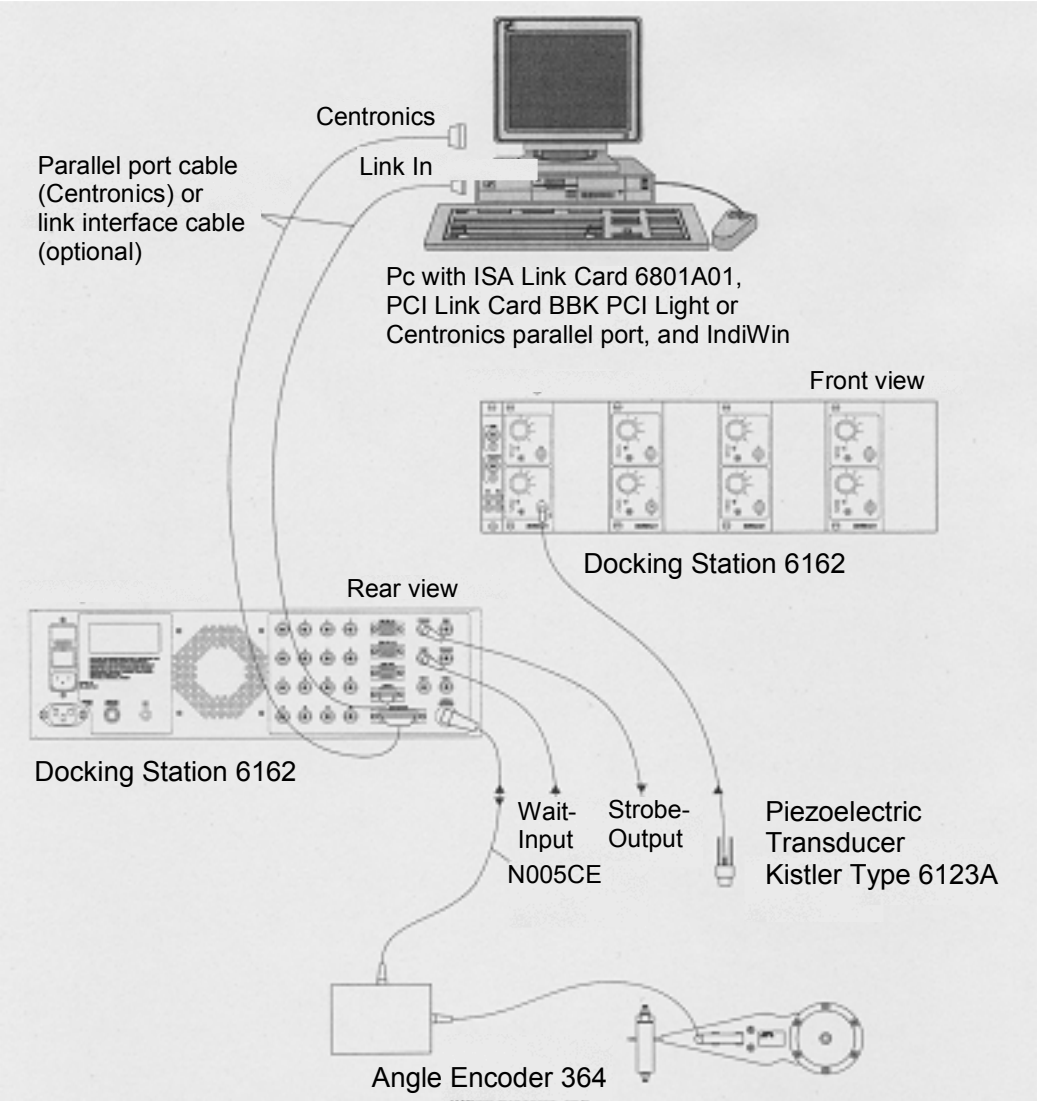
A.6 Pressure Sensor

Kistler non-cooled quartz pressure sensor 6123A.

Stainless steel body, probe diameter: 6.2 mm.

Pressure range: 0 to 250 bar.

A.7 AVL 620 Indiset Measuring Configuration



Adapted from AVL 620 Indiset Operating Instructions.

A.8 Epoxy Resin

Liquid epoxy Devcon Plastic Steel® Liquid (B).

Temperature resistance: 49°C (wet), 121°C (dry).

Thermal conductivity: 0.582 W/m K.

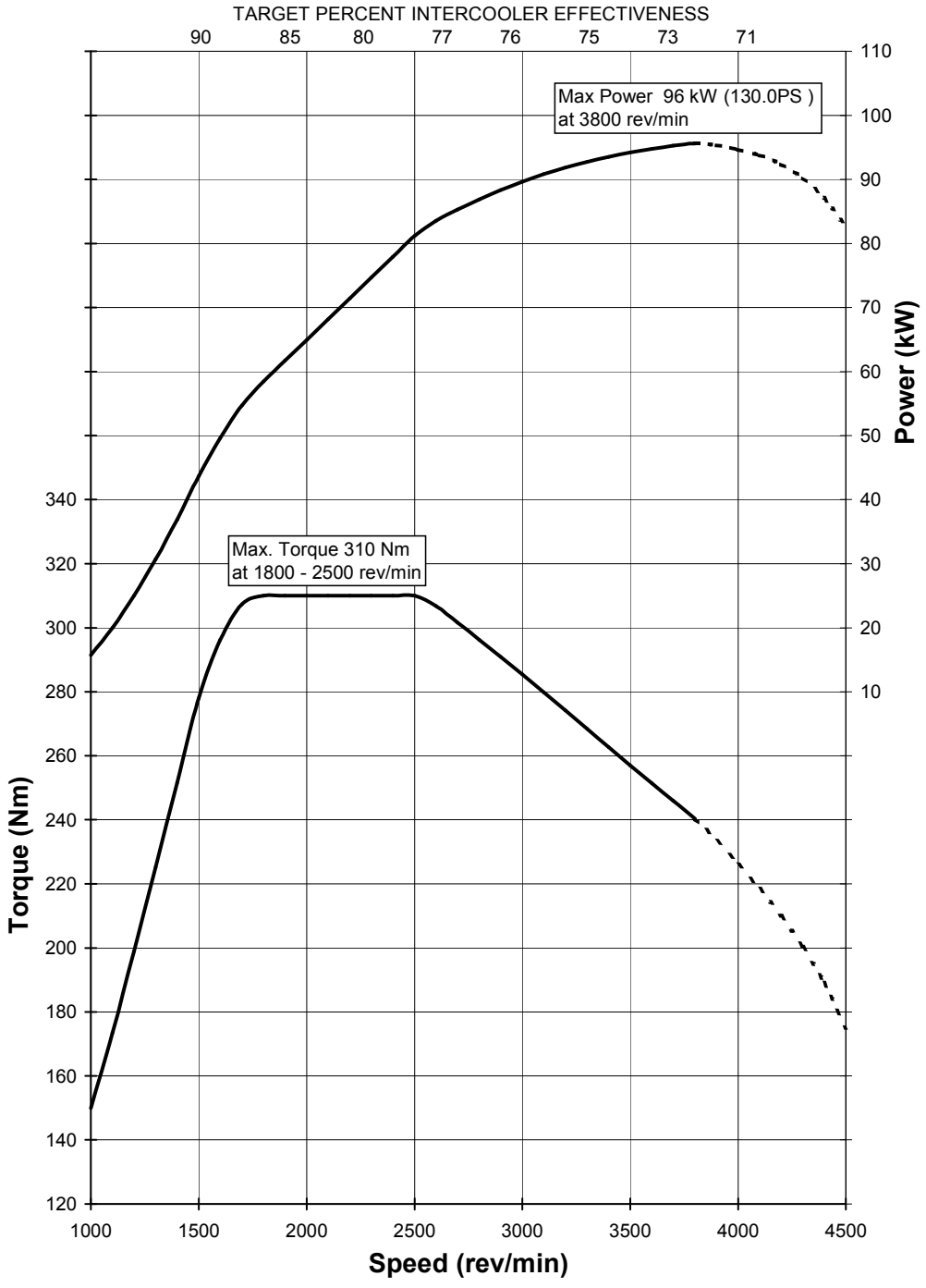
Coefficient of thermal expansion: $21 \times 10^{-6}\ K^{-1}$.

Cure time: 16 hours.

Appendix B. Engine Characteristics

B.1 Limiting Torque Curve (LTC) and Power Curve

2.OI TCI (VNT) PUMA WITH HPCR DIESEL ENGINE PERFORMANCE FOR PASSENGER CAR APPLICATION TO 88/195/EEC WITH NO VISIBLE SMOKE



Adapted from data provided by the Ford Motor Company.

B.2 Valve Data

Dimension/Material	Inlet Valve	Exhaust Valve
Valve seat angle (°)	60	45
Head diameter (mm)	29.8	25.8
Syem diameter (mm)	7	7
Length (mm)	122.2	122.1
Material	Austenitic CrNiMn steel	Austenitic CrNiMn steel

Inlet Valve			
Cam Angle (º)	Lift (mm)	Cam Angle (º)	Lift (mm)
0	0	66	8.6623
1	0.0013	67	8.6132
2	0.0063	68	8.5532
3	0.0151	69	8.4826
4	0.0276	70	8.4014
5	0.0425	71	8.3097
6	0.0576	72	8.2077
7	0.0753	73	8.0955
8	0.0996	74	7.9733
9	0.1341	75	7.8412
10	0.1811	76	7.6994
11	0.2422	77	7.5482
12	0.3178	78	7.3878
13	0.4081	79	7.2183
14	0.5131	80	7.0401
15	0.6330	81	6.8534
16	0.7678	82	6.6585
17	0.9175	83	6.4557
18	1.0822	84	6.2455
19	1.2614	85	6.0284
20	1.4545	86	5.8048
21	1.6603	87	5.5754
22	1.8777	88	5.3407
23	2.1052	89	5.1013
24	2.3412	90	4.8581
25	2.5840	91	4.6115
26	2.8320	92	4.3624
27	3.0838	93	4.1114
28	3.3378	94	3.8594
29	3.5929	95	3.6070
30	3.8480	96	3.3550
31	4.1025	97	3.1046
32	4.3556	98	2.8568
33	4.6064	99	2.6133
34	4.8544	100	2.3754
35	5.0988	101	2.1448
36	5.3390	102	1.9230
37	5.5743	103	1.7116
38	5.8042	104	1.5117
39	6.0281	105	1.3247
40	6.2454	106	1.1543
41	6.4556	107	0.9921
42	6.6584	108	0.8474
43	6.8534	109	0.7170
44	7.0401	110	0.6010

Exhaust Valve			
Cam Angle (º)	Lift (mm)	Cam Angle (º)	Lift (mm)
0	0	69	8.6811
1	0.0013	70	8.6424
2	0.0063	71	8.5951
3	0.0151	72	8.5393
4	0.0276	73	8.4751
5	0.0425	74	8.4024
6	0.0576	75	8.3213
7	0.0753	76	8.2318
8	0.1001	77	8.1341
9	0.1352	78	8.0281
10	0.1833	79	7.9140
11	0.2459	80	7.7919
12	0.3235	81	7.6617
13	0.4164	82	7.5236
14	0.5244	83	7.3777
15	0.6479	84	7.2241
16	0.7867	85	7.0628
17	0.9409	86	6.8941
18	1.1106	87	6.7180
19	1.2952	88	6.5349
20	1.4936	89	6.3448
21	1.7046	90	6.1480
22	1.9264	91	5.9448
23	2.1573	92	5.7355
24	2.3951	93	5.5205
25	2.6382	94	5.3000
26	2.8845	95	5.0746
27	3.1327	96	4.8446
28	3.3814	97	4.6106
29	3.6296	98	4.3730
30	3.8767	99	4.1326
31	4.1218	100	3.8898
32	4.3644	101	3.6455
33	4.6037	102	3.4002
34	4.8393	103	3.1548
35	5.0705	104	2.9103
36	5.2970	105	2.6681
37	5.5183	106	2.4296
38	5.7341	107	2.1968
39	5.9439	108	1.9714
40	6.1474	109	1.7553
41	6.3444	110	1.5503
42	6.5347	111	1.3578
43	6.7180	112	1.1791
44	6.8941	113	1.0150

Inlet Valve			
Cam Angle (º)	Lift (mm)	Cam Angle (º)	Lift (mm)
45	7.2183	111	0.4992
46	7.3878	112	0.4116
47	7.5482	113	0.3382
48	7.6994	114	0.2789
49	7.8412	115	0.2330
50	7.9733	116	0.1992
51	8.0955	117	0.1752
52	8.2077	118	0.1576
53	8.3097	119	0.1425
54	8.4014	120	0.1275
55	8.4826	121	0.1125
56	8.5532	122	0.0975
57	8.6132	123	0.0825
58	8.6623	124	0.0675
59	8.7006	125	0.0525
60	8.7281	126	0.0375
61	8.7445	127	0.0230
62	8.7500	128	0.0117
63	8.7445	129	0.0042
64	8.7281	130	0.0005
65	8.7006	131	0

Exhaust Valve			
Cam Angle (º)	Lift (mm)	Cam Angle (º)	Lift (mm)
45	7.0628	114	0.8658
46	7.2241	115	0.7315
47	7.3777	116	0.6120
48	7.5236	117	0.5073
49	7.6617	118	0.4172
50	7.7919	119	0.3418
51	7.9140	120	0.2810
52	8.0281	121	0.2340
53	8.1341	122	0.1996
54	8.2318	123	0.1752
55	8.3213	124	0.1576
56	8.4024	125	0.1425
57	8.4751	126	0.1275
58	8.5393	127	0.1125
59	8.5951	128	0.0975
60	8.6424	129	0.0825
61	8.6811	130	0.0675
62	8.7112	131	0.0525
63	8.7328	132	0.0375
64	8.7457	133	0.0230
65	8.7500	134	0.0117
66	8.7457	135	0.0042
67	8.7328	136	0.0005
68	8.7112	137	0

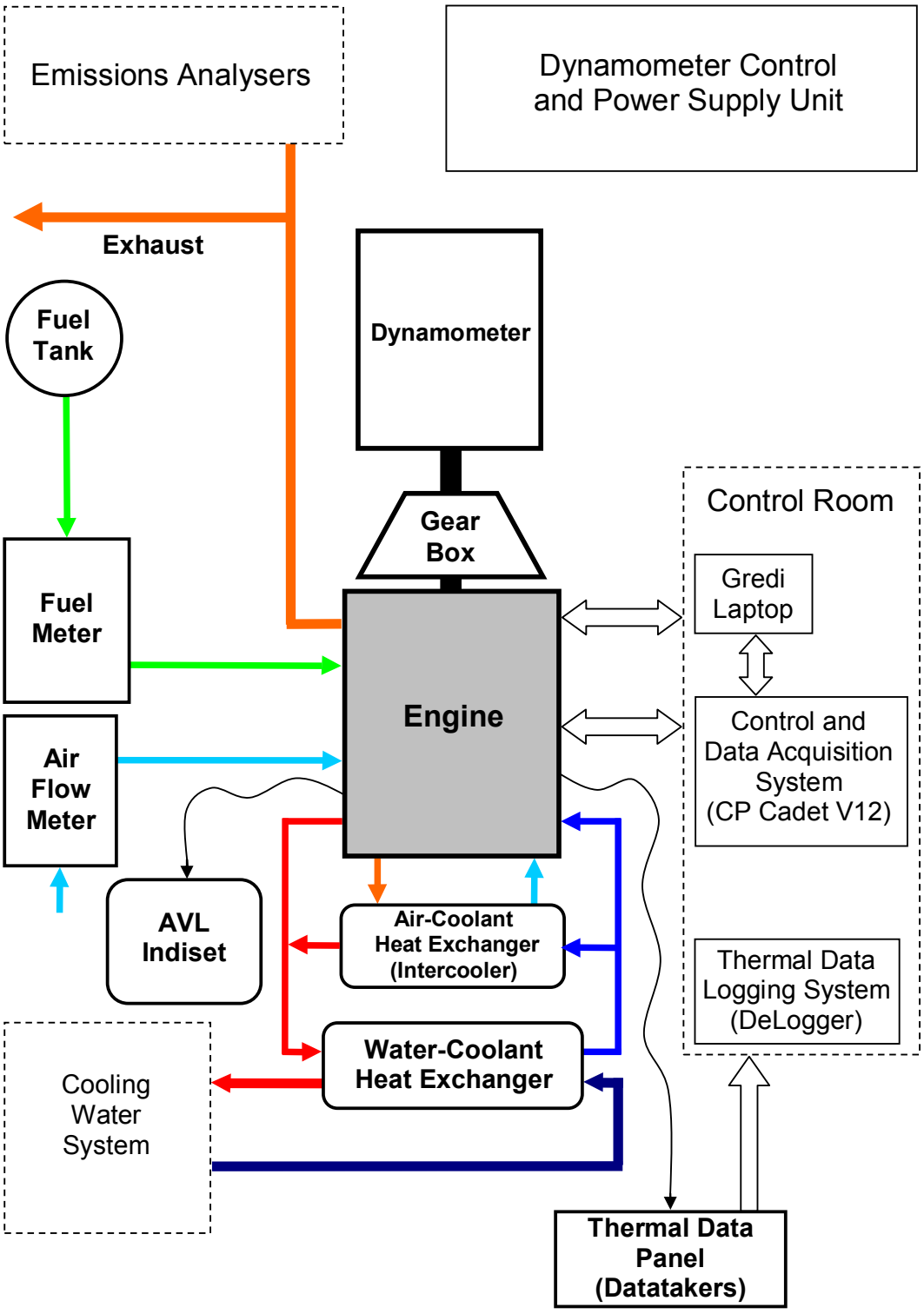
Inlet Valve		
Lift/Diameter	CDF	CDR
0	0	0
0.0364	0.1027	0.0655
0.0545	0.1880	0.1035
0.0727	0.2358	0.2065
0.0909	0.3072	0.2317
0.1091	0.3500	0.2801
0.1273	0.4008	0.3303
0.1455	0.4252	0.3696
0.1636	0.4510	0.4046
0.1818	0.4562	0.4345
0.2000	0.4557	0.4565
0.2182	0.4620	0.4809
0.2364	0.4695	0.4987
0.2545	0.4738	0.5045
0.2727	0.4784	0.5143
0.2909	0.4780	0.5201
0.3091	0.4831	0.5277
0.3927	0.4854	0.5307

Exhaust Valve		
Lift/Diameter	CDF	CDR
0	0	0
0.0394	0.1663	0.2094
0.0591	0.3206	0.3516
0.0787	0.4908	0.4972
0.0984	0.6331	0.6464
0.1181	0.7429	0.7664
0.1378	0.8566	0.9039
0.1575	1.0012	1.0248
0.1772	1.0815	1.1034
0.1969	1.1484	1.144
0.2165	1.2125	1.1894
0.2362	1.2601	1.2141
0.2559	1.2927	1.2327
0.2756	1.3151	1.2445
0.2953	1.3243	1.2518
0.3150	1.3327	1.2601
0.3347	1.3375	1.2663
0.3543	1.3398	1.2743
0.3740	1.3404	1.2794
0.3937	1.3529	1.2772

Data provided by the Ford Motor Company.

Appendix C. Test Facilities

C.1 Test Cell Schematic Layout



C.2 Test Cell Equipment

- David McClure 215kW transient AC dynamometer.
- CP Engineering Cadet V12 control and data adquisition system.
  Max 256 channels: 32 K-type thermocouples, 16 PRTs, 16 pressures and 16 analogue channels.
- Gredi engine management software.
- CP Engineering FMS1000 fuel weighing system.
- ABB Sensyflow air flow meter.
- Horiba Mexa 7000 emissions analysers.

Appendix D. Data Processing Routines

D.1 Testing Sequence

```
% This routine helps to indentify the starting and ending times
% of each test point during a testing sequence at a given speed

clc
close all
clear all
Selecttrace('D:\work\Trace files\trace20050809_4000rpm.&IT')

enginespeed = GetChannelValues('Av. Engine Speed [RPM]');
enginetorque = GetChannelValues('Av. Engine Torq [Nm]');
bothosetemp = GetChannelValues('Bottom Hose Temp [oC]');
inletairtemp = GetChannelValues(36); % Inlet air temperature
fueldemand = GetChannelValues('Fuel Demand [mg/st]');
timingdemand = GetChannelValues('Timing Demand [oBTDC]');
massairflow = GetChannelValues('Mass Air Flow [kg/hr]');

h=figure;
set(h,'PaperOrientation','landscape')
[AX,H1,H2] = plotyy(enginespeed(:,2),enginespeed(:,1),...
                    enginetorque(:,2),enginetorque(:,1));
set(H1,'LineStyle','-','Color','c');
set(H2,'LineStyle','-','Color','g');
H = [H1 H2];
legend(H,'Speed','Torque','Location','NorthWest')
set(AX(1),'YLim',[0 4500],'YTick',[0:500:4500],'YColor','k','Fontsize',20);
set(AX(2),'YLim',[-100 300],'YTick',[-100:50:300],...
          'YColor','k','Fontsize',20);
set(get(AX(1),'Ylabel'),'String','Speed (rev/min)','FontSize',22);
set(get(AX(2),'Ylabel'),'String','Torque (Nm)','FontSize',22);
xlabel('Time (s)','FontSize',22)
box off
```

D.2 Logging Times

```
% This routine helps to select the optimum loging period for each
% test point based on plots of the control variables (speed, torque,
% bottom hose coolant temperature and inlet air temperature)

clc
close all
clear all
load cpdata4000rpm
Selecttrace('D:\work\Trace files\trace20050809_4000rpm.&IT')

bothosetemp = [];
postintertemp = [];
inletairtemp = [];
enginespeed = [];
enginetorque = [];
fueldemand = [];
timingdemand = [];
% massairflow = [];
% cylheadtemp = [];
index = [];
initindex = 1;

for i = 1:length(timeset),
    enginespeed = [enginespeed; getchannelvalues('Av. Engine Speed [RPM]',...
                   timeset(i,1),timeset(i,2))];
    enginetorque = [enginetorque; getchannelvalues('Av. Engine Torq [Nm]',...
                    timeset(i,1),timeset(i,2))];
    bothosetemp = [bothosetemp; GetChannelValues('Bottom Hose Temp [oC]',...
                   timeset(i,1),timeset(i,2))];
    postintertemp=[postintertemp; GetChannelValues('Post-Intercooler [oC]',...
                   timeset(i,1),timeset(i,2))];
    inletairtemp = [inletairtemp; GetChannelValues(36,...
```

```
                    timeset(i,1),timeset(i,2))];
    fueldemand = [fueldemand; GetChannelValues('Fuel Demand [mg/st]',...
                  timeset(i,1),timeset(i,2))];
    timingdemand = [timingdemand; GetChannelValues('Timing Demand [oBTDC]',...
                    timeset(i,1),timeset(i,2))];
    index = [index; initindex, length(enginetorque)];
    initindex = length(enginetorque) + 1;
end

for i = 1:length(timeset),
    h=figure;
    set(h,'PaperOrientation','landscape')
    subplot(4,1,1)
    plot(enginespeed(index(i,1):index(i,2),2),...
         enginespeed(index(i,1):index(i,2),1),'c-')
    set(gca,'Fontsize',12);
    ylabel({'Speed';'(rev/min)'},'FontSize',16)
    xlabel('Time (s)','FontSize',16)

    subplot(4,1,2)
    plot(enginetorque(index(i,1):index(i,2),2),...
         enginetorque(index(i,1):index(i,2),1),'g-')
    set(gca,'Fontsize',12);
    ylabel({'Torque';'(Nm)'},'FontSize',16)
    xlabel('Time (s)','FontSize',16)

    subplot(4,1,3)
    plot(fueldemand(index(i,1):index(i,2),2),...
         fueldemand(index(i,1):index(i,2),1),'m-')
    set(gca,'Fontsize',12);
    ylabel({'Fuel';'Demand';'(mg/st)'},'FontSize',16)
    xlabel('Time (s)','FontSize',16)

    subplot(4,1,4)
    plot(timingdemand(index(i,1):index(i,2),2),...
         timingdemand(index(i,1):index(i,2),1),'y-')
    set(gca,'Fontsize',12);
    ylabel({'Timing';'Demand';'(oBTDC)'},'FontSize',16)
    xlabel('Time (s)','FontSize',16)
end
```

D.3 Trace File Data Stripper

```
% This routine separates the trace file from a test sequence at a given
% speed and creates data sets of average values. Also, provides the logging
% times to process the temperature data

clc
close all
clear all
load cpdata4000rpm
Selecttrace('D:\work\Trace files\trace20050809_4000rpm.&IT')

ts = tracestart;
periodms = [];
t0(1) = ts(1);
t0(2) = ts(2);
t0(3) = ts(3);
t0(4) = 0;
t0(5) = 0;
t0(6) = 0;

for i = 1:length(period),
    t0(6) = period(i);
    periodms = [periodms; datestr(datenum(t0),13)];
    t0(5) = 0;
    t0(6) = 0;
end

timeshms = [];
itimeshms = [];
```

```
etimeshms = [];
tr1 = ts;
tr2 = ts;
tss = etime(ts,t0);

times = [timeset(:,2)-period timeset(:,2)];

for i = 1:length(timeset),
    tr1(6) = ts(6) + times(i,1);
    tr2(6) = ts(6) + times(i,2);
    timeshms = [timeshms; datestr(datenum(tr1),13),'  ',...
                          datestr(datenum(tr2),13)];
    itimeshms = [itimeshms; datestr(datenum(tr1),13)];
    etimeshms = [etimeshms; datestr(datenum(tr2),13)];
end

period
periodms
times
itimeshms
etimeshms

traceheaders = str2mat(getchannel(14), getchannel(15), getchannel(34));
for i = 36:48, traceheaders = str2mat(traceheaders, getchannel(i)); end
for i = 50:52, traceheaders = str2mat(traceheaders, getchannel(i)); end
for i = 55:60, traceheaders = str2mat(traceheaders, getchannel(i)); end
traceheaders = str2mat(traceheaders, getchannel(84), getchannel(85),...
               getchannel(87), getchannel(115), getchannel(116));
for i = 118:120, traceheaders = str2mat(traceheaders, getchannel(i)); end
traceheaders = str2mat(traceheaders, getchannel(164), getchannel(165));
for i = 212:216, traceheaders = str2mat(traceheaders, getchannel(i)); end
traceheaders = str2mat(traceheaders, getchannel(220), getchannel(231),...
               getchannel(246));
for i = 260:265, traceheaders = str2mat(traceheaders, getchannel(i)); end
traceheaders = str2mat(traceheaders, getchannel(267), getchannel(268));
for i = 338:347, traceheaders = str2mat(traceheaders, getchannel(i)); end

testpoint = [];
testpointmin = [];
testpointmax = [];
testpointstd = [];
trace4000rpm = [];
trace4000rpmmin = [];
trace4000rpmmax = [];
trace4000rpmstd = [];

for i = 1:length(times),
    getchannelvalues(14);
    values = getchannelvalues(14,times(i,1),times(i,2));
    testpoint = [testpoint, mean(values(:,1))];
    testpointmin = [testpointmin, min(values(:,1))];
    testpointmax = [testpointmax, max(values(:,1))];
    testpointstd = [testpointstd, std(values(:,1))];
    getchannelvalues(15);
    values = getchannelvalues(15,times(i,1),times(i,2));
    testpoint = [testpoint, mean(values(:,1))];
    testpointmin = [testpointmin, min(values(:,1))];
    testpointmax = [testpointmax, max(values(:,1))];
    testpointstd = [testpointstd, std(values(:,1))];
    getchannelvalues(34);
    values = getchannelvalues(34,times(i,1),times(i,2));
    testpoint = [testpoint, mean(values(:,1))];
    testpointmin = [testpointmin, min(values(:,1))];
    testpointmax = [testpointmax, max(values(:,1))];
    testpointstd = [testpointstd, std(values(:,1))];
    for j = 36:48,
        getchannelvalues(j);
        values = getchannelvalues(j,times(i,1),times(i,2));
        testpoint = [testpoint, mean(values(:,1))];
        testpointmin = [testpointmin, min(values(:,1))];
```

```
    testpointmax = [testpointmax, max(values(:,1))];
    testpointstd = [testpointstd, std(values(:,1))];
end
for j = 50:52,
    getchannelvalues(j);
    values = getchannelvalues(j,times(i,1),times(i,2));
    testpoint = [testpoint, mean(values(:,1))];
    testpointmin = [testpointmin, min(values(:,1))];
    testpointmax = [testpointmax, max(values(:,1))];
    testpointstd = [testpointstd, std(values(:,1))];
end
for j = 55:60,
    getchannelvalues(j);
    values = getchannelvalues(j,times(i,1),times(i,2));
    testpoint = [testpoint, mean(values(:,1))];
    testpointmin = [testpointmin, min(values(:,1))];
    testpointmax = [testpointmax, max(values(:,1))];
    testpointstd = [testpointstd, std(values(:,1))];
end
for j = 84:85,
    getchannelvalues(j);
    values = getchannelvalues(j,times(i,1),times(i,2));
    testpoint = [testpoint, mean(values(:,1))];
    testpointmin = [testpointmin, min(values(:,1))];
    testpointmax = [testpointmax, max(values(:,1))];
    testpointstd = [testpointstd, std(values(:,1))];
end
getchannelvalues(87);
values = getchannelvalues(87,times(i,1),times(i,2));
testpoint = [testpoint, mean(values(:,1))];
testpointmin = [testpointmin, min(values(:,1))];
testpointmax = [testpointmax, max(values(:,1))];
testpointstd = [testpointstd, std(values(:,1))];
for j = 115:116,
    getchannelvalues(j);
    values = getchannelvalues(j,times(i,1),times(i,2));
    testpoint = [testpoint, mean(values(:,1))];
    testpointmin = [testpointmin, min(values(:,1))];
    testpointmax = [testpointmax, max(values(:,1))];
    testpointstd = [testpointstd, std(values(:,1))];
end
for j = 118:120,
    getchannelvalues(j);
    values = getchannelvalues(j,times(i,1),times(i,2));
    testpoint = [testpoint, mean(values(:,1))];
    testpointmin = [testpointmin, min(values(:,1))];
    testpointmax = [testpointmax, max(values(:,1))];
    testpointstd = [testpointstd, std(values(:,1))];
end
for j = 164:165,
    getchannelvalues(j);
    values = getchannelvalues(j,times(i,1),times(i,2));
    testpoint = [testpoint, mean(values(:,1))];
    testpointmin = [testpointmin, min(values(:,1))];
    testpointmax = [testpointmax, max(values(:,1))];
    testpointstd = [testpointstd, std(values(:,1))];
end
for j = 212:216,
    getchannelvalues(j);
    values = getchannelvalues(j,times(i,1),times(i,2));
    testpoint = [testpoint, mean(values(:,1))];
    testpointmin = [testpointmin, min(values(:,1))];
    testpointmax = [testpointmax, max(values(:,1))];
    testpointstd = [testpointstd, std(values(:,1))];
end
getchannelvalues(220);
values = getchannelvalues(220,times(i,1),times(i,2));
testpoint = [testpoint, mean(values(:,1))];
testpointmin = [testpointmin, min(values(:,1))];
testpointmax = [testpointmax, max(values(:,1))];
```

```
    testpointstd = [testpointstd, std(values(:,1))];
    getchannelvalues(231);
    values = getchannelvalues(231,times(i,1),times(i,2));
    testpoint = [testpoint, mean(values(:,1))];
    testpointmin = [testpointmin, min(values(:,1))];
    testpointmax = [testpointmax, max(values(:,1))];
    testpointstd = [testpointstd, std(values(:,1))];
    getchannelvalues(246);
    values = getchannelvalues(246,times(i,1),times(i,2));
    testpoint = [testpoint, mean(values(:,1))];
    testpointmin = [testpointmin, min(values(:,1))];
    testpointmax = [testpointmax, max(values(:,1))];
    testpointstd = [testpointstd, std(values(:,1))];
    for j = 260:265,
        getchannelvalues(j);
        values = getchannelvalues(j,times(i,1),times(i,2));
        testpoint = [testpoint, mean(values(:,1))];
        testpointmin = [testpointmin, min(values(:,1))];
        testpointmax = [testpointmax, max(values(:,1))];
        testpointstd = [testpointstd, std(values(:,1))];
    end
    for j = 267:268,
        getchannelvalues(j);
        values = getchannelvalues(j,times(i,1),times(i,2));
        testpoint = [testpoint, mean(values(:,1))];
        testpointmin = [testpointmin, min(values(:,1))];
        testpointmax = [testpointmax, max(values(:,1))];
        testpointstd = [testpointstd, std(values(:,1))];
    end
    for j = 338:347,
        getchannelvalues(j);
        values = getchannelvalues(j,times(i,1),times(i,2));
        testpoint = [testpoint, mean(values(:,1))];
        testpointmin = [testpointmin, min(values(:,1))];
        testpointmax = [testpointmax, max(values(:,1))];
        testpointstd = [testpointstd, std(values(:,1))];
    end
    trace4000rpm = [trace4000rpm; testpoint];
    trace4000rpmmin = [trace4000rpmmin; testpointmin];
    trace4000rpmmax = [trace4000rpmmax; testpointmax];
    trace4000rpmstd = [trace4000rpmstd; testpointstd];
    testpoint = [];
    testpointmin = [];
    testpointmax = [];
    testpointstd = [];
    pause(1)
end

motorrow = trace4000rpm(2,:);
trace4000rpm(2,:) = trace4000rpm(1,:);
trace4000rpm(1,:) = motorrow;

motorrow = trace4000rpmmin(2,:);
trace4000rpmmin(2,:) = trace4000rpmmin(1,:);
trace4000rpmmin(1,:) = motorrow;

motorrow = trace4000rpmmax(2,:);
trace4000rpmmax(2,:) = trace4000rpmmax(1,:);
trace4000rpmmax(1,:) = motorrow;

motorrow = trace4000rpmstd(2,:);
trace4000rpmstd(2,:) = trace4000rpmstd(1,:);
trace4000rpmstd(1,:) = motorrow;

for i = 1:length(times),
    h=figure;
    set(h,'PaperOrientation','landscape')
    getchannelvalues(15);
    values = GetChannelValues('Av. Engine Speed [RPM]',times(i,1),times(i,2));
    subplot(4,1,1)
```

```
    plot(values(:,2),values(:,1),'c-')
    set(gca,'Fontsize',12);
    ylabel({'Speed';'(rpm)'},'FontSize',16)
    xlabel('Time (s)','FontSize',16)

    getchannelvalues(14);
    values = GetChannelValues('Av. Engine Torq [Nm]',times(i,1),times(i,2));
    subplot(4,1,2)
    plot(values(:,2),values(:,1),'g-')
    set(gca,'Fontsize',12);
    ylabel({'Torque';'(Nm)'},'FontSize',16)
    xlabel('Time (s)','FontSize',16)

    getchannelvalues(34);
    values = GetChannelValues('Bottom Hose Temp [oC]',times(i,1),times(i,2));
    subplot(4,1,3)
    plot(values(:,2),values(:,1),'b-')
    set(gca,'Fontsize',12);
    ylabel({'Bottom Hose';'Temperature';'(°C)'},'FontSize',16)
    xlabel('Time (s)','FontSize',16)

    getchannelvalues(36);
    values = GetChannelValues(36,times(i,1),times(i,2));
    subplot(4,1,4)
    plot(values(:,2),values(:,1),'r-')
    set(gca,'Fontsize',12);
    ylabel({'Inlet Air';'Temperature';'(°C)'},'FontSize',16)
    xlabel('Time (s)','FontSize',16)
end
```

D.4 Temperature Data Stripper

```
Sub Macro1()

' Macro1 Macro
' Macro recorded 30/09/2005 by Carlos A. Finol

' % This routine separates a temperature file from a test sequence at a given
' % speed and creates a data set for each test point (engine torque condition)

' This bit set final time and intital counters
    Sheets("Time").Select
    Range("B11").Select
    finaltime = ActiveCell.Value
    currentline = 1
    pasteline = 1
    sheetno = 1
    logtime = 1
    Range("C1").Select
    blanks = ActiveCell.Value

Do
' This bit initializes starting and ending times
    Sheets("Time").Select
    Range("A" & logtime).Select
    starttime = ActiveCell.Value
    Range("B" & logtime).Select
    endtime = ActiveCell.Value
    Sheets.Add
    Sheets("Template_Prog_Temp").Select
    Range("B" & currentline).Select
    currenttime = ActiveCell.Value

    Do While currenttime <= endtime And currenttime <> blanks

' These are the warming-up and extra lines
        If currenttime < starttime Then GoTo 10

' These are the line to be copied
        If currenttime <= endtime Then
            Range("A" & currentline, "ag" & currentline).Select
```

```
            Selection.Copy
            Sheets("Sheet" & sheetno).Select
            Range("A" & pasteline).Select
            ActiveSheet.Paste
            pasteline = pasteline + 1
        End If

10      currentline = currentline + 1
        Sheets("Template_Prog_Temp").Select
        Range("B" & currentline).Select
        currenttime = ActiveCell.Value
    Loop

' This bit starts a new sheet
    pasteline = 1
    sheetno = sheetno + 1
    logtime = logtime + 1
    pasteline = 1
    Sheets("Template_Prog_Temp").Select
    Range("B" & currentline).Select
    currenttime = ActiveCell.Value

Loop While currenttime <= finaltime And currenttime <> blanks

' This bit organizes things
    Sheets("Template_Prog_Temp").Select
    Range("A1").Select
    Application.CutCopyMode = False
    Sheets("Sheet1").Select
    Sheets("Sheet1").Name = "10%LTC"
    Sheets("Sheet2").Select
    Sheets("Sheet2").Name = "Motored"
    Sheets("Motored").Select
    Sheets("Motored").Move Before:=Sheets(1)
    Sheets("Sheet3").Select
    Sheets("Sheet3").Name = "20%LTC"
    Sheets("Sheet4").Select
    Sheets("Sheet4").Name = "30%LTC"
    Sheets("Sheet5").Select
    Sheets("Sheet5").Name = "40%LTC"
    Sheets("Sheet6").Select
    Sheets("Sheet6").Name = "50%LTC"
    Sheets("Sheet7").Select
    Sheets("Sheet7").Name = "60%LTC"
    Sheets("Sheet8").Select
    Sheets("Sheet8").Name = "70%LTC"
    Sheets("Sheet9").Select
    Sheets("Sheet9").Name = "80%LTC"
    Sheets("Sheet10").Select
    Sheets("Sheet10").Name = "90%LTC"
    Sheets("Sheet11").Select
    Sheets("Sheet11").Name = "100%LTC"
    Sheets(Array("Time", "Template_Prog_Temp")).Select
    Sheets("Template_Prog_Temp").Activate
    Sheets(Array("Time", "Template_Prog_Temp")).Move Before:=Sheets(1)
    Sheets("Template_Prog_Temp").Select
    Range("A1").Select

' Plots of Temperatue vs Real time
    Sheets("Motored").Select
    Range("B3").Select
    Charts.Add
    ActiveChart.ChartType = xlXYScatterSmoothNoMarkers
    ActiveChart.SetSourceData
Source:=Sheets("Motored").Range("B1:B200,F1:F200"), _
        PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:="Motored"
    With ActiveChart
        .HasTitle = False
        .Axes(xlCategory, xlPrimary).HasTitle = True
```

```
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Real time"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Temperature"
    End With
    With ActiveChart.Axes(xlCategory)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    With ActiveChart.Axes(xlValue)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    ActiveChart.HasLegend = False
    ActiveSheet.Shapes("Chart 1").IncrementLeft 54.75
    ActiveSheet.Shapes("Chart 1").IncrementTop -95.25
    ActiveChart.Axes(xlCategory).Select
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.NumberFormat = "0.0"
    ActiveChart.ChartArea.Select
    ActiveSheet.Shapes("Chart 1").ScaleWidth 1.04, msoFalse,
msoScaleFromTopLeft
    ActiveSheet.Shapes("Chart 1").ScaleHeight 1.11, msoFalse,
msoScaleFromTopLeft
    ActiveWindow.Visible = False
    Windows("Template_Prog_by_torque.xls").Activate
    Range("A1").Select
    Selection.Columns.AutoFit

    Sheets("10%LTC").Select
    Range("B3").Select
    Charts.Add
    ActiveChart.ChartType = xlXYScatterSmoothNoMarkers
    ActiveChart.SetSourceData
Source:=Sheets("10%LTC").Range("B1:B200,F1:F200"), _
        PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:="10%LTC"
    With ActiveChart
        .HasTitle = False
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Real time"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Temperature"
    End With
    With ActiveChart.Axes(xlCategory)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    With ActiveChart.Axes(xlValue)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    ActiveChart.HasLegend = False
    ActiveSheet.Shapes("Chart 1").IncrementLeft 54.75
    ActiveSheet.Shapes("Chart 1").IncrementTop -95.25
    ActiveChart.Axes(xlCategory).Select
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.NumberFormat = "0.0"
    ActiveChart.ChartArea.Select
    ActiveSheet.Shapes("Chart 1").ScaleWidth 1.04, msoFalse,
msoScaleFromTopLeft
    ActiveSheet.Shapes("Chart 1").ScaleHeight 1.11, msoFalse,
msoScaleFromTopLeft
    ActiveWindow.Visible = False
    Windows("Template_Prog_by_torque.xls").Activate
    Range("A1").Select
    Selection.Columns.AutoFit

    Sheets("20%LTC").Select
    Range("B3").Select
    Charts.Add
```

```
    ActiveChart.ChartType = xlXYScatterSmoothNoMarkers
    ActiveChart.SetSourceData
Source:=Sheets("20%LTC").Range("B1:B200,F1:F200"), _
        PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:="20%LTC"
    With ActiveChart
        .HasTitle = False
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Real time"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Temperature"
    End With
    With ActiveChart.Axes(xlCategory)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    With ActiveChart.Axes(xlValue)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    ActiveChart.HasLegend = False
    ActiveSheet.Shapes("Chart 1").IncrementLeft 54.75
    ActiveSheet.Shapes("Chart 1").IncrementTop -95.25
    ActiveChart.Axes(xlCategory).Select
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.NumberFormat = "0.0"
    ActiveChart.ChartArea.Select
    ActiveSheet.Shapes("Chart 1").ScaleWidth 1.04, msoFalse,
msoScaleFromTopLeft
    ActiveSheet.Shapes("Chart 1").ScaleHeight 1.11, msoFalse,
msoScaleFromTopLeft
    ActiveWindow.Visible = False
    Windows("Template_Prog_by_torque.xls").Activate
    Range("A1").Select
    Selection.Columns.AutoFit

    Sheets("30%LTC").Select
    Range("B3").Select
    Charts.Add
    ActiveChart.ChartType = xlXYScatterSmoothNoMarkers
    ActiveChart.SetSourceData
Source:=Sheets("30%LTC").Range("B1:B200,F1:F200"), _
        PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:="30%LTC"
    With ActiveChart
        .HasTitle = False
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Real time"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Temperature"
    End With
    With ActiveChart.Axes(xlCategory)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    With ActiveChart.Axes(xlValue)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    ActiveChart.HasLegend = False
    ActiveSheet.Shapes("Chart 1").IncrementLeft 54.75
    ActiveSheet.Shapes("Chart 1").IncrementTop -95.25
    ActiveChart.Axes(xlCategory).Select
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.NumberFormat = "0.0"
    ActiveChart.ChartArea.Select
    ActiveSheet.Shapes("Chart 1").ScaleWidth 1.04, msoFalse,
msoScaleFromTopLeft
    ActiveSheet.Shapes("Chart 1").ScaleHeight 1.11, msoFalse,
msoScaleFromTopLeft
```

```
    ActiveWindow.Visible = False
    Windows("Template_Prog_by_torque.xls").Activate
    Range("A1").Select
    Selection.Columns.AutoFit

    Sheets("40%LTC").Select
    Range("B3").Select
    Charts.Add
    ActiveChart.ChartType = xlXYScatterSmoothNoMarkers
    ActiveChart.SetSourceData
Source:=Sheets("40%LTC").Range("B1:B200,F1:F200"), _
        PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:="40%LTC"
    With ActiveChart
        .HasTitle = False
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Real time"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Temperature"
    End With
    With ActiveChart.Axes(xlCategory)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    With ActiveChart.Axes(xlValue)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    ActiveChart.HasLegend = False
    ActiveSheet.Shapes("Chart 1").IncrementLeft 54.75
    ActiveSheet.Shapes("Chart 1").IncrementTop -95.25
    ActiveChart.Axes(xlCategory).Select
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.NumberFormat = "0.0"
    ActiveChart.ChartArea.Select
    ActiveSheet.Shapes("Chart 1").ScaleWidth 1.04, msoFalse,
msoScaleFromTopLeft
    ActiveSheet.Shapes("Chart 1").ScaleHeight 1.11, msoFalse,
msoScaleFromTopLeft
    ActiveWindow.Visible = False
    Windows("Template_Prog_by_torque.xls").Activate
    Range("A1").Select
    Selection.Columns.AutoFit

    Sheets("50%LTC").Select
    Range("B3").Select
    Charts.Add
    ActiveChart.ChartType = xlXYScatterSmoothNoMarkers
    ActiveChart.SetSourceData
Source:=Sheets("50%LTC").Range("B1:B200,F1:F200"), _
        PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:="50%LTC"
    With ActiveChart
        .HasTitle = False
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Real time"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Temperature"
    End With
    With ActiveChart.Axes(xlCategory)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    With ActiveChart.Axes(xlValue)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    ActiveChart.HasLegend = False
    ActiveSheet.Shapes("Chart 1").IncrementLeft 54.75
    ActiveSheet.Shapes("Chart 1").IncrementTop -95.25
```

```
    ActiveChart.Axes(xlCategory).Select
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.NumberFormat = "0.0"
    ActiveChart.ChartArea.Select
    ActiveSheet.Shapes("Chart 1").ScaleWidth 1.04, msoFalse,
msoScaleFromTopLeft
    ActiveSheet.Shapes("Chart 1").ScaleHeight 1.11, msoFalse,
msoScaleFromTopLeft
    ActiveWindow.Visible = False
    Windows("Template_Prog_by_torque.xls").Activate
    Range("A1").Select
    Selection.Columns.AutoFit

    Sheets("60%LTC").Select
    Range("B3").Select
    Charts.Add
    ActiveChart.ChartType = xlXYScatterSmoothNoMarkers
    ActiveChart.SetSourceData
Source:=Sheets("60%LTC").Range("B1:B200,F1:F200"), _
        PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:="60%LTC"
    With ActiveChart
        .HasTitle = False
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Real time"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Temperature"
    End With
    With ActiveChart.Axes(xlCategory)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    With ActiveChart.Axes(xlValue)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    ActiveChart.HasLegend = False
    ActiveSheet.Shapes("Chart 1").IncrementLeft 54.75
    ActiveSheet.Shapes("Chart 1").IncrementTop -95.25
    ActiveChart.Axes(xlCategory).Select
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.NumberFormat = "0.0"
    ActiveChart.ChartArea.Select
    ActiveSheet.Shapes("Chart 1").ScaleWidth 1.04, msoFalse,
msoScaleFromTopLeft
    ActiveSheet.Shapes("Chart 1").ScaleHeight 1.11, msoFalse,
msoScaleFromTopLeft
    ActiveWindow.Visible = False
    Windows("Template_Prog_by_torque.xls").Activate
    Range("A1").Select
    Selection.Columns.AutoFit

    Sheets("70%LTC").Select
    Range("B3").Select
    Charts.Add
    ActiveChart.ChartType = xlXYScatterSmoothNoMarkers
    ActiveChart.SetSourceData
Source:=Sheets("70%LTC").Range("B1:B200,F1:F200"), _
        PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:="70%LTC"
    With ActiveChart
        .HasTitle = False
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Real time"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Temperature"
    End With
    With ActiveChart.Axes(xlCategory)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
```

```
    End With
    With ActiveChart.Axes(xlValue)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    ActiveChart.HasLegend = False
    ActiveSheet.Shapes("Chart 1").IncrementLeft 54.75
    ActiveSheet.Shapes("Chart 1").IncrementTop -95.25
    ActiveChart.Axes(xlCategory).Select
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.NumberFormat = "0.0"
    ActiveChart.ChartArea.Select
    ActiveSheet.Shapes("Chart 1").ScaleWidth 1.04, msoFalse,
msoScaleFromTopLeft
    ActiveSheet.Shapes("Chart 1").ScaleHeight 1.11, msoFalse,
msoScaleFromTopLeft
    ActiveWindow.Visible = False
    Windows("Template_Prog_by_torque.xls").Activate
    Range("A1").Select
    Selection.Columns.AutoFit

    Sheets("80%LTC").Select
    Range("B3").Select
    Charts.Add
    ActiveChart.ChartType = xlXYScatterSmoothNoMarkers
    ActiveChart.SetSourceData
Source:=Sheets("80%LTC").Range("B1:B200,F1:F200"), _
        PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:="80%LTC"
    With ActiveChart
        .HasTitle = False
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Real time"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Temperature"
    End With
    With ActiveChart.Axes(xlCategory)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    With ActiveChart.Axes(xlValue)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    ActiveChart.HasLegend = False
    ActiveSheet.Shapes("Chart 1").IncrementLeft 54.75
    ActiveSheet.Shapes("Chart 1").IncrementTop -95.25
    ActiveChart.Axes(xlCategory).Select
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.NumberFormat = "0.0"
    ActiveChart.ChartArea.Select
    ActiveSheet.Shapes("Chart 1").ScaleWidth 1.04, msoFalse,
msoScaleFromTopLeft
    ActiveSheet.Shapes("Chart 1").ScaleHeight 1.11, msoFalse,
msoScaleFromTopLeft
    ActiveWindow.Visible = False
    Windows("Template_Prog_by_torque.xls").Activate
    Range("A1").Select
    Selection.Columns.AutoFit

    Sheets("90%LTC").Select
    Range("B3").Select
    Charts.Add
    ActiveChart.ChartType = xlXYScatterSmoothNoMarkers
    ActiveChart.SetSourceData
Source:=Sheets("90%LTC").Range("B1:B200,F1:F200"), _
        PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:="90%LTC"
    With ActiveChart
        .HasTitle = False
```

```
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Real time"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Temperature"
    End With
    With ActiveChart.Axes(xlCategory)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    With ActiveChart.Axes(xlValue)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    ActiveChart.HasLegend = False
    ActiveSheet.Shapes("Chart 1").IncrementLeft 54.75
    ActiveSheet.Shapes("Chart 1").IncrementTop -95.25
    ActiveChart.Axes(xlCategory).Select
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.NumberFormat = "0.0"
    ActiveChart.ChartArea.Select
    ActiveSheet.Shapes("Chart 1").ScaleWidth 1.04, msoFalse,
msoScaleFromTopLeft
    ActiveSheet.Shapes("Chart 1").ScaleHeight 1.11, msoFalse,
msoScaleFromTopLeft
    ActiveWindow.Visible = False
    Windows("Template_Prog_by_torque.xls").Activate
    Range("A1").Select
    Selection.Columns.AutoFit

    Sheets("100%LTC").Select
    Range("B3").Select
    Charts.Add
    ActiveChart.ChartType = xlXYScatterSmoothNoMarkers
    ActiveChart.SetSourceData
Source:=Sheets("100%LTC").Range("B1:B200,F1:F200"), _
        PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:="100%LTC"
    With ActiveChart
        .HasTitle = False
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Real time"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Temperature"
    End With
    With ActiveChart.Axes(xlCategory)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    With ActiveChart.Axes(xlValue)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    ActiveChart.HasLegend = False
    ActiveSheet.Shapes("Chart 1").IncrementLeft 54.75
    ActiveSheet.Shapes("Chart 1").IncrementTop -95.25
    ActiveChart.Axes(xlCategory).Select
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.NumberFormat = "0.0"
    ActiveChart.ChartArea.Select
    ActiveSheet.Shapes("Chart 1").ScaleWidth 1.04, msoFalse,
msoScaleFromTopLeft
    ActiveSheet.Shapes("Chart 1").ScaleHeight 1.11, msoFalse,
msoScaleFromTopLeft
    ActiveWindow.Visible = False
    Windows("Template_Prog_by_torque.xls").Activate
    Range("A1").Select
    Selection.Columns.AutoFit
    ActiveWindow.ScrollWorkbookTabs Position:=xlFirst
    Sheets("100%LTC").Select
    Columns("C:C").Select
```

```
    Application.CutCopyMode = False
    Selection.Delete Shift:=xlToLeft
    Sheets("90%LTC").Select
    Columns("C:C").Select
    Selection.Delete Shift:=xlToLeft
    Sheets("80%LTC").Select
    Columns("C:C").Select
    Selection.Delete Shift:=xlToLeft
    Sheets("70%LTC").Select
    Columns("C:C").Select
    Selection.Delete Shift:=xlToLeft
    Sheets("60%LTC").Select
    Columns("C:C").Select
    Selection.Delete Shift:=xlToLeft
    Sheets("50%LTC").Select
    Columns("C:C").Select
    Selection.Delete Shift:=xlToLeft
    Sheets("40%LTC").Select
    Columns("C:C").Select
    Selection.Delete Shift:=xlToLeft
    Sheets("30%LTC").Select
    Columns("C:C").Select
    Selection.Delete Shift:=xlToLeft
    Sheets("20%LTC").Select
    Columns("C:C").Select
    Selection.Delete Shift:=xlToLeft
    Sheets("10%LTC").Select
    Columns("C:C").Select
    Selection.Delete Shift:=xlToLeft
    Sheets("Motored").Select
    Columns("C:C").Select
    Selection.Delete Shift:=xlToLeft
    Range("A1").Select
    Sheets("10%LTC").Select
    Range("A1").Select
    Sheets("20%LTC").Select
    Range("A1").Select
    Sheets("30%LTC").Select
    Range("A1").Select
    Sheets("40%LTC").Select
    Range("A1").Select
    Sheets("50%LTC").Select
    Range("A1").Select
    Sheets("60%LTC").Select
    Range("A1").Select
    Sheets("70%LTC").Select
    Range("A1").Select
    Sheets("80%LTC").Select
    Range("A1").Select
    Sheets("90%LTC").Select
    Range("A1").Select
    Sheets("100%LTC").Select
    Range("A1").Select
End Sub
```

D.5 Thermal Gradients, Wall Surface Temperature and Heat Flux in the Engine Block

```
% This routines calculates the the thermal gradient and the gas-side wall
% surface temperature and heat flux at locations in the engine block

clc
clear all
close all
load test4000rpm
[m,n]=size(blocktemp4000rpm);

% Thermal gradient, temperature and heat flux calculations
blockgrad4000rpm = [];
twg4000rpm = [];
```

```
errtwg = [];
errtwc = [];

for i=1:m,
    twg = [];
    grad = [];
    j = 1;
       j1 = j+2;
    while j<=n,
        if j <10 | (j >15 & j <28) | (j >30 & j <46) | (j >72 & j < 79) |...
           j > 87
            j1 = j+2;
            js = 1;
            r = [45,46.5,48];
            rws = [43,45,46.5,48,50];
        elseif (j >9 & j <16) | (j >45 & j <49) | (j >78 & j <82)
            j1 = j+1;
            js = 1;
            r = [45,46.5];
            rws = [43,45,46.5,48,50];
        elseif  (j >54 & j <58) | (j >60 & j <73)
            j1 = j+2;
            js = 2;
            r = [45,48];
            rws = [43,45,46.5,48,50];
        end
        [pr,E]=polyfit(log(r/rws(1)),blocktemp4000rpm(i,j:js:j1),1);
        grad = [grad, pr(1)];
        [yr,err]=polyval(pr,log(rws/rws(1)),E);
        twg = [twg, yr(1)];
        if j == 46 | j == 79
            j = j+9;
        elseif j == 25 | j == 55 | j == 70
            j = j+6;
        else
            j = j+3;
        end
    end
    blockgrad4000rpm = [blockgrad4000rpm; grad];
    twg4000rpm = [twg4000rpm; twg];
end
blockheatflux4000rpm = -45*blockgrad4000rpm/43;     % kW/m2

locname = ['Gas side wall temperature (Location A, 4000 rev/min)';...
           'Gas side wall temperature (Location B, 4000 rev/min)';...
           'Gas side wall temperature (Location C, 4000 rev/min)';...
           'Gas side wall temperature (Location D, 4000 rev/min)';...
           'Gas side wall temperature (Location E, 4000 rev/min)';...
           'Gas side wall temperature (Location F, 4000 rev/min)';...
           'Gas side wall temperature (Location G, 4000 rev/min)';...
           'Gas side wall temperature (Location I, 4000 rev/min)';...
           'Gas side wall temperature (Location J, 4000 rev/min)'];

titletext = ['Location A';'Location B';'Location C';...
             'Location D';'Location E';'Location F';...
             'Location G';'Location I';'Location J'];

legendtext = ['Motored ';'10% LTC ';'20% LTC ';'30% LTC ';'40% LTC ';...
              '50% LTC ';'60% LTC ';'70% LTC ';'80% LTC ';'90% LTC ';...
              '100% LTC'];

format1 = ['k-';'y-';'c-';'b-';'m-';'g-';'r-';'b:';'m:';'g:';'r:'];
format1l = ['-';'-';'-';'-';'-';'-';'-';':';':';':';':'];
format1c = ['k';'y';'c';'b';'m';'g';'r';'b';'m';'g';'r'];
formatc = [[0 0 .5];[1 1 0];[.7 .7 0];[0 .7 .7];[.7 0 0];[0 0 1];...
           [0 1 0];[0 1 1];[1 0 1];[1 .4 0];[1 0 0]];

% Plots predicted temperatures
y = [10 25 40 55 70 85 100];
for k=1:9,
```

```
    h(k)=figure;
    if k == 1
        location = y;
        intervalues = linspace(10,100,90);
        measvalues = twg4000rpm(:,1:7);
    elseif k == 2
        location = [10 25 55];
        intervalues = linspace(10,55,45);
        measvalues = twg4000rpm(:,8:10);
    elseif k == 3
        location = y(1:2);
        intervalues = y(1:2);
        measvalues = twg4000rpm(:,11:12);
    elseif k == 4
        location = y(1:2);
        intervalues = y(1:2);
        measvalues = twg4000rpm(:,13:14);
    elseif k == 5
        location = y(2);
        intervalues = y(2);
        measvalues = twg4000rpm(:,15);
    elseif k == 6
        location = [40 70 85 100];
        intervalues = linspace(40,100,60);
        measvalues = twg4000rpm(:,16:19);
    elseif k == 7
        location = [10 40 55];
        intervalues = linspace(10,55,45);
        measvalues = twg4000rpm(:,20:22);
    elseif k == 8
        location = y(1);
        intervalues = y(1);
        measvalues = twg4000rpm(:,23);
    else
        location = y(1);
        intervalues = y(1);
        measvalues = twg4000rpm(:,24);
    end

    for i = 1:m,
        if k == 5 | k > 7
            plotvalues = measvalues(i,:);
        else
            plotvalues =
interp1(location,measvalues(i,:),intervalues,'pchip');
        end
        figure(h(k))
        hold on
        plot(intervalues,plotvalues,format1(i,:))
    end
    grid
    grid on
    set(h(k),'NumberTitle','off','Name',locname(k,:),...
             'PaperOrientation','landscape',...
             'PaperPosition',[0.6345 1.129 28.41 18.73],...
             'PaperSize',[29.68 20.98]);

% Adjust colors
    for i=1:m
            set(findobj(gca,'Type','line','LineStyle',format1l(i,:),...
                            'Color',format1c(i,:)),'LineStyle','--',...
                            'Color',formatc(i,:))
    end
    set(gca,'YLim',[90 170],'XLim',[0 110])
    set(gca,'XTick',[0:11:110],'Fontsize',22)
    ylabel({'Wall Surface Temperature (°C)'},'FontSize',28)
    xlabel('Distance from top deck (mm)','FontSize',28)

% Plots measured values
    for i = 1:m,
```

```
        figure(h(k))
        hold on
        plot(location,measvalues(i,:),'o','MarkerSize',9,...
            'MarkerEdgeColor',formatc(i,:),'MarkerFaceColor',formatc(i,:))
    end
    title(titletext(k,:),'Fontsize',30)
    lh=legend(legendtext(1:m,:),'Location','NorthEast');
    legend(lh,{legendtext(1:m,:)},'FontSize',16,'location','NorthEast')
end
lasth = h(k);

% Plots predicted heat fluxes
locname = ['Heat flux (Location A, 4000 rev/min)';...
           'Heat flux (Location B, 4000 rev/min)';...
           'Heat flux (Location C, 4000 rev/min)';...
           'Heat flux (Location D, 4000 rev/min)';...
           'Heat flux (Location E, 4000 rev/min)';...
           'Heat flux (Location F, 4000 rev/min)';...
           'Heat flux (Location G, 4000 rev/min)';...
           'Heat flux (Location I, 4000 rev/min)';...
           'Heat flux (Location J, 4000 rev/min)'];

for k=1:9,
    h(k+lasth)=figure;
    if k == 1
        location = y;                                % A1 to A7
        intervalues = linspace(10,100,90);
        measvalues = blockheatflux4000rpm(:,1:7);
    elseif k == 2
        location = [10 25 55];                       % B1, B2 and B4
        intervalues = linspace(10,55,45);
        measvalues = blockheatflux4000rpm(:,8:10);
    elseif k == 3
        location = y(1:2);                           % C1 and C2
        intervalues = y(1:2);
        measvalues = blockheatflux4000rpm(:,11:12);
    elseif k == 4
        location = y(1:2);                           % D1 and D2
        intervalues = y(1:2);
        measvalues = blockheatflux4000rpm(:,13:14);
    elseif k == 5
        location = y(2);                             % E2
        intervalues = y(2);
        measvalues = blockheatflux4000rpm(:,15);
    elseif k == 6
        location = [40 70 85 100];                   % F3, F5, F6 and F7
        intervalues = linspace(40,100,60);
        measvalues = blockheatflux4000rpm(:,16:19);
    elseif k == 7
        location = [10 40 55];                       % G1, G3 and G4
        intervalues = linspace(10,55,45);
        measvalues = blockheatflux4000rpm(:,20:22);
    elseif k == 8
        location = y(1);                             % I1
        intervalues = y(1);
        measvalues = blockheatflux4000rpm(:,23);
    else
        location = y(1);                             % J1
        intervalues = y(1);
        measvalues = blockheatflux4000rpm(:,24);
    end

    for i = 1:m,
        if k == 5 | k > 7
            plotvalues = measvalues(i,:);
        else
            plotvalues = interp1(location,measvalues(i,:),...
                                 intervalues,'pchip');
        end
        figure(h(k+lasth))
```

```
        hold on
        plot(intervalues,plotvalues,format1(i,:))
    end
    grid on
    set(h(k+lasth),'NumberTitle','off','Name',locname(k,:),...
                   'PaperOrientation','landscape',...
                   'PaperPosition',[0.6345 1.129 28.41 18.73],...
                   'PaperSize',[29.68 20.98]);

% Adjust colors
    for i=1:m
            set(findobj(gca,'Type','line','LineStyle',format1l(i,:),...
                            'Color',format1c(i,:)),'LineStyle','--',...
                            'Color',formatc(i,:))
    end
    set(gca,'XLim',[0 110],'YLim',[0 300])
    set(gca,'XTick',[0:10:110],'Fontsize',22)
    set(gca,'YTick',[0:50:300],'Fontsize',22)
    ylabel({'{Heat Flux (kW/m}^{2})'},'FontSize',28)
    xlabel('Distance from top deck (mm)','FontSize',28)

% Plots measured values
    for i=1:m,
        figure(h(k+lasth))
        hold on
        plot(location,measvalues(i,:),'o','MarkerSize',9,....
            'MarkerEdgeColor',formatc(i,:),'MarkerFaceColor',formatc(i,:))
    end
    lh=legend(legendtext(1:m,:),'Location','NorthEast');
    legend(lh,{legendtext(1:m,:)},'FontSize',16,'location','NorthEast')
end
```

D.6 Thermal Gradients, Wall Surface Temperature and Heat Flux in the Cylinder Head

```
% This routines calculates the the thermal gradient and the gas-side wall
% surface temperature and heat flux at locations in the cylinder head

clc
clear all
close all
load test4000rpm
[m,n]=size(headtemp4000rpm);

theta = [153 153 153 153 153 153 245 245 354 354 353 353 74 74 74 74];
theta = pi/180*theta;

rho = [15;17;22.5;24.5;34.5;36.5;22.5;24.5;22.5;24.5;33.5;35.3;15;17;...
       22.5;24.5];

torque = trace4000rpm(2:m,1);

% Thermal gradient, temperature and heat flux calculations
for i=1:m,
    for k=1:8,
        if k==1 | k==2
            radialgrad7mm(i,k) = (headtemp4000rpm(i,4) - ...
                                  headtemp4000rpm(i,2))/7.5;
            headtemp7mmadj(i,k) = headtemp4000rpm(i,2*k) - ...
                                  2*radialgrad7mm(i,k);
        elseif k==3
            radialgrad7mm(i,k) = (headtemp4000rpm(i,6) - ...
                                  headtemp4000rpm(i,4))/12;
            headtemp7mmadj(i,k) = headtemp4000rpm(i,6) - ...
                                  2*radialgrad7mm(i,k);
        elseif k>3 & k<7
            radialgrad7mm(i,k) = (headtemp4000rpm(i,12) - ...
                                  headtemp4000rpm(i,10))/10.8;
            if k==6
                headtemp7mmadj(i,k) = headtemp4000rpm(i,2*k) - ...
```

```
                                    1.8*radialgrad7mm(i,k);
            else
                headtemp7mmadj(i,k) = headtemp4000rpm(i,2*k) - ...
                                      2*radialgrad7mm(i,k);
            end
        else
            radialgrad7mm(i,k) = (headtemp4000rpm(i,16) - ...
                                  headtemp4000rpm(i,14))/7.5;
            headtemp7mmadj(i,k) = headtemp4000rpm(i,2*k) - ...
                                  2*radialgrad7mm(i,k);
        end
    end
    k=1;
    for j=1:2:n-1,
        headgrad4000rpm(i,k) = 0.2*(headtemp4000rpm(i,j) - ...
                                    headtemp7mmadj(i,k));
        temphs4000rpm(i,k) = headtemp4000rpm(i,j) + 2*headgrad4000rpm(i,k);
        k = k+1;
    end
end
headheatflux4000rpm = 195*headgrad4000rpm;
[m,n]=size(temphs4000rpm);

% Plots predicted temperatures
figname = ['Head surface temperature (Locations A, B, C and D at 4000rpm)';...
           'Head surface temperature (Locations E, F, G and H at 4000rpm)'];

locname = ['Head Surface Temperature (Locations A at 4000rpm)';...
           'Head Surface Temperature (Locations B at 4000rpm)';...
           'Head Surface Temperature (Locations C at 4000rpm)';...
           'Head Surface Temperature (Locations D at 4000rpm)';...
           'Head Surface Temperature (Locations E at 4000rpm)';...
           'Head Surface Temperature (Locations F at 4000rpm)';...
           'Head Surface Temperature (Locations G at 4000rpm)';...
           'Head Surface Temperature (Locations H at 4000rpm)'];

title1 = ['Set A';'Set B';'Set C';'Set D';'Set E';'Set F';'Set G';'Set H'];
title2 = [' Exhaust-inlet valve bridge ';'  Inlet-inlet valve bridge  ';...
          ' Inlet-exhaust valve bridge ';'Exhaust-exhaust valve bridge'];
title4 = ['Locations A, B and C';'     Location D     ';
          ' Locations E and F  ';' Locations G and H  '];
formatc = [[0 0 .5];[1 1 0];[.7 .7 0];[0 .7 .7];[.7 0 0];[0 0 1];[0 1 0];...
           [0 1 1];[1 0 1];[1 .4 0];[1 0 0]];
format = ['ro';'ms';'b^';'gd';'cp';'y+';'kh';'rx';'r>';'m*';'b<'];
format1 = ['ro';'ms';'b^';'ms';'ms';'b^';'ro';'ms'];

k=1;
for j=1:n,
    if j == 1 | j == 4 | j == 5 | j == 7
        fp=figure;
        set(fp,'NumberTitle','off','Name',...
               'Cylinder head surface temperature vs Torque at 4000rpm',...
               'PaperOrientation','landscape')
    end
    plot(torque,temphs4000rpm(2:m,j),format1(j,:),...
                'MarkerSize',10,'MarkerFaceColor',format1(j,1))
    hold on

    if j == 3 | j == 4 | j == 6 | j == 8
        grid on
        ylabel({'Wall Surface Temperature (°C)'},'FontSize',28)
        xlabel('Torque (Nm)','FontSize',28)
        set(gca,'XLim',[0 320],'YLim',[100 200])
        set(gca,'XTick',[0 40 80 120 160 200 240 280 320],'FontSize',22)
        title({title2(k,:);'Speed =4000rpm'},'Fontsize',26)
        if j == 3
            legend(strcat('r=',num2str(rho(1:2:5)),'mm'),2)
        elseif j == 4
            legend(strcat('r=',num2str(rho(7)),'mm'),2)
        else
```

```
            legend(strcat('r=',num2str(rho(j*2-3:2:j*2-1)),'mm'),2)
        end
        hold off
        k = k+1;
    end
end

k=1;
for i=2:m,
    figure(5)
    plot(rho(1:2:5),temphs4000rpm(i,1:3),format(k,:),...
                    'LineStyle','none','Color',formatc(k,:),...
                    'MarkerSize',10,'MarkerEdgeColor',formatc(k,:),...
                    'MarkerFaceColor',formatc(k,:))
    hold on

    figure(6)
    plot(rho(7),temphs4000rpm(i,4),format(k,:),...
               'LineStyle','none','Color',formatc(k,:),...
               'MarkerSize',10,'MarkerEdgeColor',formatc(k,:),...
               'MarkerFaceColor',formatc(k,:))
    hold on

    figure(7)
    plot(rho(9:2:11),temphs4000rpm(i,5:6),format(k,:),...
                    'LineStyle','none','Color',formatc(k,:),...
                    'MarkerSize',10,'MarkerEdgeColor',formatc(k,:),...
                    'MarkerFaceColor',formatc(k,:))
    hold on

    figure(8)
    plot(rho(13:2:15),temphs4000rpm(i,7:8),format(k,:),...
                     'LineStyle','none','Color',formatc(k,:),...
                     'MarkerSize',10,'MarkerEdgeColor',formatc(k,:),...
                     'MarkerFaceColor',formatc(k,:))
    hold on
    k = k+1;
end

for l=1:4,
    fp=figure(4+l);
    set(fp,'NumberTitle','off','Name',...
          'Cylinder head surface temperature vs Chamber radius at 4000rpm',...
          'PaperOrientation','landscape')
    grid on
    ylabel({'Head surface temperature (°C)';'gas side      '},'FontSize',24)
    xlabel('Radius (mm)','FontSize',24)
    set(gca,'XLim',[0 43],'YLim',[100 200])
    set(gca,'XTick',[0 5 10 15 20 25 30 35 40],'FontSize',22)
    title({title2(l,:);'Speed =4000rpm'},'Fontsize',26)
    legend('10%LTC','20%LTC','30%LTC','40%LTC','50%LTC','60%LTC',...
           '70%LTC','80%LTC','90%LTC','100%LTC',2)
end
lastfp = fp;

% Plots predicted heat fluxes
figname = ['Head heat flux(Locations A, B, C and D at 4000rpm)';...
           'Head heat flux(Locations E, F, G and H at 4000rpm)'];

locname = ['Head heat flux(Locations A at 4000rpm)';...
           'Head heat flux(Locations B at 4000rpm)';...
           'Head heat flux(Locations C at 4000rpm)';...
           'Head heat flux(Locations D at 4000rpm)';...
           'Head heat flux(Locations E at 4000rpm)';...
           'Head heat flux(Locations F at 4000rpm)';...
           'Head heat flux(Locations G at 4000rpm)';...
           'Head heat flux(Locations H at 4000rpm)'];
k=1;
for j=1:n,
    if j == 1 | j == 4 | j == 5 | j == 7
```

```
        fp=figure(lastfp+k);
        set(fp,'NumberTitle','off',...
               'Name','Cylinder head heat flux vs Torque at 4000rpm',...
               'PaperOrientation','landscape')
    end
    plot(torque,headheatflux4000rpm(2:m,j),format1(j,:),'MarkerSize',6,...
                        'MarkerFaceColor',format1(j,1))
    hold on
    if j == 3 | j == 4 | j == 6 | j == 8
        grid on
        ylabel('{Heat flux (kW/m}^{2})','FontSize',22)
        xlabel('Torque (Nm)','FontSize',22)
        set(gca,'XLim',[0 320],'YLim',[0 800])
        set(gca,'XTick',[0 40 80 120 160 200 240 280 320],'FontSize',18)
        title({title2(k,:);'Speed =4000rpm'},'Fontsize',24)
        if j == 3
            legend(strcat('r=',num2str(rho(1:2:5)),'mm'),2)
        elseif j == 4
            legend(strcat('r=',num2str(rho(7)),'mm'),2)
        else
            legend(strcat('r=',num2str(rho(j*2-3:2:j*2-1)),'mm'),2)
        end
        hold off
        k = k+1;
    end
end

k=1;
for i=2:m,
    figure(lastfp+5)
    plot(rho(1:2:5),headheatflux4000rpm(i,1:3),format(k,:),...
                   'LineStyle','none','Color',formatc(k,:),...
                   'MarkerSize',6,'MarkerEdgeColor',formatc(k,:),...
                   'MarkerFaceColor',formatc(k,:))
    hold on

    figure(lastfp+6)
    plot(rho(7),headheatflux4000rpm(i,4),format(k,:),...
               'LineStyle','none','Color',formatc(k,:),...
               'MarkerSize',6,'MarkerEdgeColor',formatc(k,:),...
               'MarkerFaceColor',formatc(k,:))
    hold on

    figure(lastfp+7)
    plot(rho(9:2:11),headheatflux4000rpm(i,5:6),format(k,:),...
                    'LineStyle','none','Color',formatc(k,:),...
                    'MarkerSize',6,'MarkerEdgeColor',formatc(k,:),...
                    'MarkerFaceColor',formatc(k,:))
    hold on

    figure(lastfp+8)
    plot(rho(13:2:15),headheatflux4000rpm(i,7:8),format(k,:),...
                     'LineStyle','none','Color',formatc(k,:),...
                     'MarkerSize',6,'MarkerEdgeColor',formatc(k,:),...
                     'MarkerFaceColor',formatc(k,:))
    hold on
    k = k+1;
end

for l=1:4,
    fp=figure(lastfp+4+l);
    set(fp,'NumberTitle','off',...
           'Name','Cylinder head heat flux vs Chamber radius at 4000rpm',...
           'PaperOrientation','landscape')
    grid on
    ylabel('{Heat flux (kW/m}^{2})','FontSize',22)
    xlabel('Radius (mm)','FontSize',22)
    set(gca,'XLim',[0 43],'YLim',[0 1000])
    set(gca,'XTick',[0 5 10 15 20 25 30 35 40],'FontSize',18)
    title({title2(l,:);'Speed =4000rpm'},'Fontsize',24)
```

```
    set(gca,'FontSize',12)
    legend('10%LTC','20%LTC','30%LTC','40%LTC','50%LTC','60%LTC',...
           '70%LTC','80%LTC','90%LTC','100%LTC',2)
end
```

D.7 Gas Temperature

```
% Calculates gas temperature (K) at 4000 rev/min

clc
clear all
close all

load 'test4000rpm'
load 'incyldata_4000rpm'
load 'intake_valves_lift'
load 'exhaust_valves_lift'
load 'air_properties'

trace = trace4000rpm(11,:);  % 100% LTC data
enginesp = trace(2); % Engine speed (rev/min)
inmantemp = trace(4)+273.15; % Inlet manifold pressure (K)
preturtemp = trace(9)+273.15; % Pre-turbine temperature (K)
posturtemp = trace(10)+273.15; % Post-turbine temperature (K)
barompress = trace(29); % Barometric pressure (kPa)
airmassflow = trace(28); % Intake air mass flow rate (kg/hr)
fuelmassflow = trace(34); % Fuel mass flow rate (g/s)
exmassflow = trace(42); % Exhaust gas mass flow rate (kg/hr)
percentEGR = trace(51); % EGR percentage
injdegBTDC = trace(57); % Ijection timing (°BTDC)
tabletemp = air_properties(:,1); % K
tableCp = air_properties(:,2); % J/kgK
inmanpress = 0.01*(trace(31) + barompress); % Inlet Manifold Pressure (bar)
% Cylinder pressure at 100% LTC (bar)
cylpress = [incyldata_4000rpm(:,42);incyldata_4000rpm(1,42)];
% Cylinder pressure at motored conditions (bar)
motpress = [incyldata_4000rpm(:,2);incyldata_4000rpm(1,2)];

% Engine data
bore = 86; % mm
stroke = 86; % mm
crankr = 43; % mm
conrod = 160; % mm
compresR = 19;
conrodcrankr = conrod/crankr;
crankrconrod = crankr/conrod;
cylarea = 0.01*pi*bore^2/4; % cm^2
displvol = 0.1*stroke*cylarea; % cm^3
clearvol = displvol/(compresR-1); % cm^3

% Valve dimensions (mm)
invalveDv = 29.8;
invalveDs = 7;
inletDp = 24.1;
inletDm = 27.7;
inseatW = 2.1;
exvalveDv = 25.8;
exvalveDs = 7;
exhaustDp = 21.1;
exhaustDm = 23.9;
exseatW = 1.9;
invalveBeta = 60*pi/180; % degrees
exvalveBeta = 45*pi/180; % degrees

% Crank angle
degCA = [0:0.1:720];
radCA = pi*degCA/180;

% Filters measured cylinder gas pressure
a = 11;
b = ones(1,a);
```

```
cylpressf = filtfilt(b,a,cylpress);
motpressf = filtfilt(b,a,motpress);

% Calculates mean piston speed (m/s)
meansp = (0.002/60)*stroke*enginesp;

% Exhaust manifold pressure (bar)
constantR = 287; % J/kgK
preturbCp = interp1(tabletemp,tableCp,preturtemp); % J/kgK
gasgamma = preturbCp/(preturbCp-constantR);
preturpress = 0.01*barompress*((preturtemp)/(posturtemp))^...
                              (gasgamma/(gasgamma-1));

% Calculates time step for 0.1 degCA at the given speed
timestep = 60*0.1/(360*enginesp);

% Calculates fuel mass (g) per cycle per cylinder at 4000rpm
fuelmass = 2*60*fuelmassflow/(4*enginesp);

% Calculates gas mass (g) per cycle per cylinder at 4000rpm
exgasmass = 2*60*1000*exmassflow/(3600*4*enginesp);

% Calculates air mass (g) per cycle per cylinder at 4000rpm
inairmass = 2*60*1000*airmassflow/(3600*4*enginesp);
ingasmass = inairmass*(1+percentEGR/100); % Air mass + EGR fraction

% Calculates instantaneous cylinder volume (cm^3)
for i = 1:length(radCA),
    if i == 1
        instheight(i) = 0.01;
    else
        instheight(i) = crankr*(conrodcrankr+1-cos(radCA(i))-...
                        sqrt(conrodcrankr^2-sin(radCA(i))^2));
        if instheight(i) < 0
            instheight(i) = 0;
        end
    end
    instvol(i) = clearvol+0.1*cylarea*instheight(i);

% Sets valves opening events
    if degCA(i) == 480     % Exhaust valves start to open
        evo = i;
    elseif degCA(i) == 697 % Intake valves start to open
        ivo = i;
    end
end

% Sets valve lift (mm) on a crank angle basis
k = 1;
for i = 1:length(invalve_lift),
    if i == length(invalve_lift)
        degcamshaft(k) = invalve_lift(i,1);
    else
        degcamshaft(k) = invalve_lift(i,1);
        degcamshaft(k+1) = 0.5*(invalve_lift(i,1)+invalve_lift(i+1,1));
        k = k+2;
    end
end
invalveliftcam = interp1(invalve_lift(:,1),invalve_lift(:,2),degcamshaft);

k = 1;
for i = 1:length(exvalve_lift),
    if i == length(exvalve_lift)
        degcamshaft(k) = exvalve_lift(i,1);
    else
        degcamshaft(k) = exvalve_lift(i,1);
        degcamshaft(k+1) = 0.5*(exvalve_lift(i,1)+exvalve_lift(i+1,1));
        k = k+2;
    end
end
```

```
exvalveliftcam = interp1(exvalve_lift(:,1),exvalve_lift(:,2),degcamshaft);

j = ivo;
k = 1;
inliftCA = [];
invaluesCA = [];
while k <= length(invalveliftcam),
    if j > 7201
        j = j-7200;
    end
    inliftCA = [inliftCA invalveliftcam(k)];
    invaluesCA = [invaluesCA degCA(j)];

    if degCA(j) == 720
        inliftCA = [inliftCA invalveliftcam(k)];
        invaluesCA = [invaluesCA 0];
    end
    k = k+1;
    j = j+10;
end

j = evo;
k = 1;
exliftCA = [];
exvaluesCA = [];
while k <= length(exvalveliftcam),
    if j > 7201
        j = j-7200;
    end
    exliftCA = [exliftCA exvalveliftcam(k)];
    exvaluesCA = [exvaluesCA degCA(j)];

    if degCA(j) == 720
        exliftCA = [exliftCA exvalveliftcam(k)];
        exvaluesCA = [exvaluesCA 0];
    end
    k = k+1;
    j = j+10;
end

% Sets valves closing events
ivcCA = degCA(ivo)+length(inliftCA)-1;
if ivcCA > 720
    ivcCA = ivcCA-720;
end
evcCA = degCA(evo)+length(exliftCA)-1;

if evcCA > 720
    evcCA = evcCA-720;
end

for i = 1:length(radCA),
    if degCA(i) == evcCA     % Exhaust valve fully closed
        evc = i;
    elseif degCA(i) == ivcCA % Intake valve fully closed
        ivc = i;
    end
end

% Calculates valves L/D ratio, CDF and CDR
invalveliftCA = interp1(invaluesCA,inliftCA,degCA);
invalratioLD = invalveliftCA/invalveDv;
invalveCDF = interp1(invalve_coeff(:,1),invalve_coeff(:,2),invalratioLD);
invalveCDR = interp1(invalve_coeff(:,1),invalve_coeff(:,3),invalratioLD);

exvalveliftCA = interp1(exvaluesCA,exliftCA,degCA);
exvalratioLD = exvalveliftCA/exvalveDv;
exvalveCDF = interp1(exvalve_coeff(:,1),exvalve_coeff(:,2),exvalratioLD);
exvalveCDR = interp1(exvalve_coeff(:,1),exvalve_coeff(:,3),exvalratioLD);
```

```
% Calculates intake minimum area (mm^2)
factorinA = 1.0034;
infactor1 = inseatW/(sin(invalveBeta)*cos(invalveBeta));
infactor2 = sqrt((inletDp^2-invalveDs^2)/(4*inletDm)-inseatW^2) + ...
                  inseatW*tan(invalveBeta);

for i = 1:length(radCA),
    if infactor1 > invalveliftCA(i) & invalveliftCA(i) > 0
        invalveAm(i) = pi*invalveliftCA(i)*cos(invalveBeta)*...
                       (invalveDv - 2*inseatW + invalveliftCA(i)*...
                        sin(2*invalveBeta)/2);

    elseif infactor2 >= invalveliftCA(i) & invalveliftCA(i) > infactor1
        invalveAm(i) = pi*inletDm*sqrt((invalveliftCA(i)-inseatW*...
                        tan(invalveBeta))^2+inseatW^2);

    elseif invalveliftCA(i) > infactor2
        invalveAm(i) = pi*(inletDp^2-invalveDs^2)/4;
    else
        invalveAm(i) = 0;
    end
    invalveAr(i) = factorinA*invalveAm(i);
end

% Calculates exhaust minimum area (mm^2)
factorexA = 1.019;
exfactor1 = exseatW/(sin(exvalveBeta)*cos(exvalveBeta));
exfactor2 = sqrt((exhaustDp^2-exvalveDs^2)/(4*exhaustDm)-exseatW^2)+...
                  exseatW*tan(exvalveBeta);

for i = 1:length(radCA),
    if exfactor1 > exvalveliftCA(i) & exvalveliftCA(i) > 0
        exvalveAm(i) = pi*exvalveliftCA(i)*cos(exvalveBeta)*...
                       (exvalveDv-2*exseatW + exvalveliftCA(i)*...
                        sin(2*exvalveBeta)/2);

    elseif exfactor2 >= exvalveliftCA(i) & exvalveliftCA(i) > exfactor1
        exvalveAm(i) = pi*exhaustDm*sqrt((exvalveliftCA(i)-...
        exseatW*tan(exvalveBeta))^2+exseatW^2);

    elseif exvalveliftCA(i) > exfactor2
        exvalveAm(i) = pi*(exhaustDp^2-exvalveDs^2)/4;
    else
        exvalveAm(i) = 0;
    end
    exvalveAr(i) = factorexA*exvalveAm(i);
end

% Calculates total gas mass (g) after injection
% and gas temperature (K) from IVC to EVO
injCA = 3600-int16(injdegBTDC);
totalinmass = ingasmass;

for i = ivc:evo,
    if i == injCA
        totalinmass = ingasmass+fuelmass;

    insttemp(i) = 100*cylpressf(i)*instvol(i)/(totalinmass*constantR);
    cylmass(i) = totalinmass;
end

% Calculates gas temperature (K) from IVO to IVC
i = ivc-1;
j = 7200;
while i+j > ivo,
    if i < 1
        i = i+7200;
        j = 0;
    end
```

```
    if i == 7200
        l = 1;
    else
        l = i+1;
    end
    tol = 1;
    tm = insttemp(l);

    while abs(tol) > 0.001
        Cp = interp1(tabletemp,tableCp,tm);
        airgamma = Cp/(Cp-constantR);
        tf = insttemp(l)*(cylpressf(l)/cylpressf(i))^((1-airgamma)/...
                          airgamma);
        tol = tm-tf;
        tm = (tm+tf)/2;
    end
    insttemp(i) = tf;
    i = i-1;
end

% Calculates gas mass (g) through intake valves
i = ivo;
j = 7200;
cylmass(ivo-1) = 0;
while i <= j+ivc,
    if i > 7200
        i = i-7200;
        j = 0;
    end

    if i == 1
        l = 7200;
    else
        l = i-1;
    end

% Forward flow
    if inmanpress > cylpressf(i)

% Flow is not choked
        if cylpressf(i)/inmanpress > (2*airgamma/(airgamma+1))^...
                                        (airgamma/(airgamma-1))

            inmass(i) = 200*timestep*(invalveCDF(i)*invalveAr(i)*...
                            inmanpress/sqrt(constantR*(inmantemp)))*...
                          ((cylpressf(i)/inmanpress)^(1/airgamma))*...
                            sqrt((2*airgamma/(airgamma-1))*...
                         (((cylpressf(i)/inmanpress)^...
                          ((airgamma-1)/airgamma))-1));
% Flow is choked
        else
            inmass(i) = 200*timestep*(invalveCDF(i)*invalveAr(i)*...
                            inmanpress/sqrt(constantR*(inmantemp)))*...
                            sqrt(airgamma)*(2/(airgamma+1))^...
                          ((airgamma+1)/(2*(airgamma-1)));
        end

% Reverse flow
    elseif inmanpress < cylpressf(i)

% Flow is not choked
        if inmanpress/cylpressf(i) > (2*airgamma/(airgamma+1))^...
                                        (airgamma/(airgamma-1))

            inmass(i) =-200*timestep*(invalveCDR(i)*invalveAr(i)*...
                            cylpressf(i)/sqrt(constantR*(inmantemp)))*...
                          ((inmanpress/cylpressf(i))^(1/airgamma))*...
                            sqrt((2*airgamma/(airgamma-1))*...
                         (((inmanpress/cylpressf(i))^...
                          ((airgamma-1)/airgamma))-1));
```

```
% Flow is choked
        else
            inmass(i) =-200*timestep*(invalveCDR(i)*invalveAr(i)*...
                            cylpressf(i)/sqrt(constantR*(inmantemp)))*...
                            sqrt(airgamma)*(2/(airgamma+1))^...
                          ((airgamma+1)/(2*(airgamma-1)));
        end
    else
        inmass(i) = 0;
    end
    cylmass(i) = cylmass(l)+inmass(i);
    i = i+1;
end
totalinmass = cylmass(ivc);

% Adjusts mass in the cylinder after fuel injection
for i = ivc:evo,
    if i == injCA
        totalinmass = totalinmass+fuelmass;
    end
    cylmass(i) = totalinmass;
end

% Calculates gas mass (g) through exhaust valves
% and temperature (K) from EVO to EVC
i = evo;
j = 7200;
k = 0;
k1 = 0;
while i <= evc-1+j,
    if i > 7200
        i = i-7200;
        j = 0;
        k = 7200;
    end

    if i == 1
        l = 7200;
    else
        l = i-1;
    end
    tol = 1;
    tm = insttemp(l);

    while abs(tol) > 0.001
        Cp = interp1(tabletemp,tableCp,tm);
        gasgamma = Cp/(Cp-constantR);
        tf = insttemp(l)*(cylpressf(i)/cylpressf(l))^...
                        ((gasgamma-1)/gasgamma);
        tol = tf-tm;
        tm = 0.5*(tm+tf);
    end

% Adjusts temperature from IVO to EVC
    if i+k >= ivo & inmass(l) > 0 & inmass(i) > 0 & ...
                  cylmass(l) > 0 & cylmass(i) > 0
        if k1 == 0
            ifac = l;
            k1 = 1;
        end
        massfactor(i) = sum(inmass(ifac:l))/cylmass(l);
        insttemp(i) = (1-massfactor(i))*tf+massfactor(i)*insttemp(evc);
    else
        insttemp(i) = tf;
    end

% Forward flow
    if cylpressf(i) > preturpress

% Flow is not choked
```

```
        if preturpress/cylpressf(i) > (2*gasgamma/(gasgamma+1))^...
                                        (gasgamma/(gasgamma-1))

            exmass(i) = 200*timestep*(exvalveCDF(i)*exvalveAr(i)*...
                          cylpressf(i)/sqrt(constantR*insttemp(i)))*...
                        ((preturpress/cylpressf(i))^(1/gasgamma))*...
                          sqrt((2*gasgamma/(gasgamma-1))*...
                         (1-((preturpress/cylpressf(i))^...
                        ((gasgamma-1)/gasgamma))));
% Flow is choked
        else
            exmass(i) = 200*timestep*(exvalveCDF(i)* ...
                          exvalveAr(i)*cylpressf(i)/...
                          sqrt(constantR*insttemp(i)))*...
                          sqrt(gasgamma)*(2/(gasgamma+1))^...
                          ((gasgamma+1)/(2*(gasgamma-1)));
        end

% Reverse flow
    elseif cylpressf(i) < preturpress

% Flow is not choked
        if cylpressf(i)/preturpress > (2*gasgamma/(gasgamma+1))^...
                                        (gasgamma/(gasgamma-1))

            exmass(i) =-200*timestep*(exvalveCDR(i)*exvalveAr(i)*...
                          preturpress/sqrt(constantR*insttemp(i)))*...
                        ((cylpressf(i)/preturpress)^(1/gasgamma))*...
                          sqrt((2*gasgamma/(gasgamma-1))*...
                         (1-((cylpressf(i)/preturpress)^...
                        ((gasgamma-1)/gasgamma))));
% Flow is choked
        else
            exmass(i) =-200*timestep*(exvalveCDR(i)*exvalveAr(i)*...
                          preturpress/sqrt(constantR*insttemp(i)))*...
                          sqrt(gasgamma)*(2/(gasgamma+1))^...
                        ((gasgamma+1)/(2*(gasgamma-1)));
        end
    else
        exmass(i) = 0;
    end

    if i+k >= ivo
        cylmass(i) = cylmass(l)-exmass(i)+inmass(i);
    else
        cylmass(i) = cylmass(l)-exmass(i);
    end
    i = i+1;
end
totalexmass = sum(exmass);

% Calculates gas density (kg/m3)
for i=1:7200,
    gasdensity(i) = 1e3*cylmass(i)/instvol(i);
end

% Plots gas temperature
h=figure;
    set(h,'NumberTitle','on','PaperOrientation','landscape',...
           'PaperPosition',[0.6345 1.129 28.41 18.73],...
           'PaperSize',[29.68 20.98])
    hold on
    plot(degCA(1:7200),insttemp(1:7200),'r','LineWidth',1.5)
    plot(degCA(evc):degCA(evc),0:2500,'k',degCA(ivc):degCA(ivc),0:2500,'k',...
         degCA(evo):degCA(evo),0:2500,'k',degCA(ivo):degCA(ivo),0:2500,'k')
    set(gca,'Xlim',[0 720],'XTick',[0:60:720],...
            'XTickLabel',{' TDC','60','120','BDC','240','300','TDC',...
                          '420','480','BDC','600','660','TDC'},...
            'FontSize',20)
    set(gca,'Ylim',[300 2500],'YTick',[300:200:2500],...
```

```
            'YTickLabel',{[300:200:2500]},'FontSize',20)
    xlabel('Crank Angle (°)','FontSize',28)
    ylabel('Temperature (K)','FontSize',28)
    text(13,2570,'EVC','FontSize',18)
    text(227,2570,'IVC','FontSize',18)
    text(458,2570,'EVO','FontSize',18)
    text(684,2570,'IVO','FontSize',18)
    grid on
    box on
```

D.8 Heat Transfer Coefficient Correlations

```
% This routine evaluates several existing correlations and the new proposed
% equation to estimate the gas-side heat transfer coefficient

clc
clear all
close all

% Executes gas and mass temperature routine and brings geometry
run gastemp4000rpm

% Executes gas temperature airproperties routine
run air_properties

% Executes averaged heatflux and averaged head surface temperature routine
run heatflux_head4000rpm

% Executes averaged heatflux and averaged head surface temperature routine
run heatflux_block4000rpm

% Calculates averaged convective heat transfer coefficient (head)
meanheadexpercoef = 1e3*meanheadheatflux/(meangastemp - meanheadtemp);

% Calculates averaged gas temperature affecting the top thermocouples
sumtemp = [];
for i=1:7200,
    if instheight(i) >= 10
        sumtemp = [sumtemp insttemp(i)];
    end
end
meangastempblock = mean(sumtemp);

% Calculates averaged convective heat transfer coefficient (block)
meanblockexpercoef = 1e3*meanblockheatflux/(meangastempblock-meanblocktemp);

% Sets start of combustion
i = 1;
j = 0;
while i <= 7200 & j == 0
    if i > injCA & heatrelease(i) > 0
        socCA = i;
        j = 1;
    end
    i = i + 1;
end

% Eichelberg's correlation
C1 = 2.14;
for i=1:7200,
    hEichelberg(i) = C1*meansp^1/3*sqrt(cylpress(i)*insttemp(i));
end

% Annand's correlation
C1 = 0.45;
C2 = 0.7;
for i=1:7200,
    if i >= socCA  & i < evo
        C3 = 0.576;
    else
        C3 = 0;
```

```
    end
    Re(i) = 1e-3*gasdensity(i)*meansp*bore/viscosity(i);

    hAnnand(i) =1000*C1*conductivity(i)*Re(i)^C2/bore + 5.67e-8*C3*...
                    (insttemp(i)^4-meanheadtemp^4)/(insttemp(i)-meanheadtemp);
end

% Woschni's correlation
for i=1:7200,
    if i > evo | i < evc | i <= ivc | i > ivo
        C1 = 6.18; C2 = 0;
    elseif i > ivc & i < socCA
        C1 = 2.28; C2 = 0;
    else
        C1 = 2.28; C2 = 3.24e-3;
    end

    w = C1*meansp+C2*displvol*insttemp(ivc)*(cylpress(i)-motpress(i))/...
                                            (cylpress(ivc)*instvol(ivc));

    hWoschni(i) = 129.9*(1e-3*bore)^-0.2*cylpress(i)^0.8*...
                                         insttemp(i)^-0.53*w^0.8;
end

% Sitkei and Ramanaiah's correlation
C1 = 0.015;
for i=1:7200,
    instchambersurf(i) = 1e-6*pi*bore*instheight(i)+2e-6*pi*bore^2/4;
    de = 4*1e-6*instvol(i)/instchambersurf(i);

    hSitRaman(i) = 46*(1+C1)*(cylpress(i)*meansp)^0.7/(insttemp(i)^...
                   0.2*de^0.3)+5.67e-8*0.3*(insttemp(i)^4-meanheadtemp^4)/...
                                            (insttemp(i)-meanheadtemp);
end

% Hohenberg's correlation
C1 = 130; C2 = 1.4;
for i=1:7200,
    hHohenberg(i) = C1*(1e-6*instvol(i))^-0.06*cylpress(i)^0.8*...
                               insttemp(i)^-0.4*(meansp+C2)^0.8;
end

% Han, Chung, Kwon and Lee's correlation
C1 = 687;
C2 = 0.494;
C3 = 7.3e-7;
for i=1:7200,
    if i == 1
        l = 7200;
    else
        l = i-1;
    end
    Cp = specheat(i);
    gasgamma = Cp/(Cp-constantR);
    velU = C2*meansp+C3*(1e-5*gasgamma*cylpress(i)*(instvol(i)-...
                              instvol(l)) + 1e-5*instvol(i)*...
                             (cylpress(i)-cylpress(l)));

    hHanetal(i) = C1*(cylpress(i)*velU)^0.75*(1e-3*bore)^-0.25*....
                      insttemp(i)^-0.465;
end

% Proposed correlation
C1 = 0.0947;
for i=1:7200,
    Re(i) = 1e-3*gasdensity(i)*1.5*meansp*bore/viscosity(i);

    hProposed(i) =1000*C1*conductivity(i)*Re(i)^0.8/bore;
end
```

```
% Calculates cycle average coefficients
meanEichelberg = mean(hEichelberg);
meanAnnand = mean(hAnnand);
meanWoschni = mean(hWoschni);
meanSitRaman = mean(hSitRaman);
meanHohenberg = mean(hHohenberg);
meanHanetal = mean(hHanetal);
meanProposed = mean(hProposed);

cycleavg = [meanEichelberg meanAnnand meanWoschni meanSitRaman...
            meanHohenberg meanHanetal meanProposed];

hcoeffmatrix = [hEichelberg; hAnnand; hWoschni; hSitRaman; hHohenberg;...
                hHanetal];

% Plots results
formatc = [[0 0 1];[0 1 0];[1 0 0];[0 1 1];[1 .7 0];[0.502 0 0.502];...
           [0 0 0]];
h=figure;
    set(h,'NumberTitle','off','Name',...
          'Cycled-averaged heat transfer coefficient',...
          'PaperOrientation','landscape',...
          'PaperPosition',[0.6345 1.129 28.41 18.73],...
          'PaperSize',[29.68 20.98])
    for i=1:7,
        plot(cycleavg(i),i,'rs','MarkerEdgeColor',formatc(i,:),...
                          'MarkerFaceColor',formatc(i,:),'MarkerSize',9)
        hold on
    end
    plot(meanheadexpercoef,(0.5:0.001:7.5),'k','LineWidth',1)
    xlabel('{Cycle-Averaged Heat Transfer Coefficient (W/m}^{2}K)',...
           'FontSize',30)
    set(gca,'XLim',[600 1200],'XTick',[600:50:1200],...
            'XTickLabel',[600:50:1200],'FontSize',22)
    set(gca,'YLim',[0.5 7.5],'YDir','reverse')
    text(477.6,0.972,'Eichelberg','BackgroundColor',[1 1 1],'FontSize',26)
    text(479.8,1.972,'    Annand','BackgroundColor',[1 1 1],'FontSize',26)
    text(473.9,2.972,'   Woschni','BackgroundColor',[1 1 1],'FontSize',26)
    text(470.2,3.972,strvcat('Sitkei and','Ramanaiah'),...
                           'BackgroundColor',[1 1 1],'FontSize',24)
    text(481.1,4.972,strvcat('Hohenberg','modified'),...
                           'BackgroundColor',[1 1 1],'FontSize',24)
    text(488.2,5.972,'Han et al.','BackgroundColor',[1 1 1],'FontSize',26)
    text(489.2,6.972,'Proposed','BackgroundColor',[1 1 1],'FontSize',26)
    anntext = strvcat('Experimental steady-state',...
                      'heat trasnfer coefficient');
    annotation(h,'textbox','Position',[0.3633 0.4819 0.3184 0.099],...
                 'BackgroundColor',[1 1 1],'EdgeColor',[1 1 1],...
                 'FitHeightToText','off','FontName','Arial','FontSize',20,...
                 'String',anntext);
    annotation(h,'arrow',[0.5215 0.7188],[0.4759 0.3928],'LineWidth',1,...
                 'HeadStyle','plain');
    grid on
    box on

h=figure;
    set(h,'NumberTitle','off','Name','Heat transfer coefficient',...
          'PaperOrientation','landscape',...
          'PaperPosition',[0.6345 1.129 28.41 18.73],...
          'PaperSize',[29.68 20.98])
    for i=1:6,
        hold on
        plot(degCA(1:7200),hcoeffmatrix(i,:),'Color',formatc(i,:),...
                                             'LineWidth',1.5)
    end
    plot(degCA(1:7200),hHohenberg,'y-.','Color',formatc(5,:),...
                                        'LineWidth',1.5)
    plot(degCA(1:7200),hProposed,'k--','LineWidth',1.5)
    plot(degCA(evc),0:10:10000,'k',degCA(ivc),0:10:10000,'k',...
         degCA(evo),0:10:10000,'k',degCA(ivo),0:10:10000,'k')
```

```
    set(gca,'XLim',[0 720],'XTick',[0:60:720],...
            'XTickLabel',{' TDC','60','120','BDC','240','300','TDC',...
                          '420','480','BDC','600','660','TDC'},...
            'FontSize',18);
    set(gca,'YLim',[0 10000],'YTick',[0:1000:10000])
    xlabel('Crank Angle (°)','FontSize',20)
    ylabel('{Heat Transfer Coefficient (W/m}^{2}K)','FontSize',20)
    text(13,10330,'EVC','FontSize',16);
    text(227,10330,'IVC','FontSize',16);
    text(458,10330,'EVO','FontSize',16);
    text(684,10330,'IVO','FontSize',16);
    lh = legend('Eichelberg','Annand','Woschni','Sitkei and Ramanaiah',...
                'Hohenberg','Han et al.','Proposed (Equation 6.3)',1);
    legend(lh,{'Eichelberg','Annand','Woschni','Sitkei and Ramanaiah',...
               'Hohenberg','Han et al.','Hohenberg',...
               'Proposed (Equation 6.7)'},'Location','NorthWest',...
               'FontSize',14)
    grid on
    box on
```

D.9 Piston Model

```
% This routines calculates the heat transferred from piston rings and skirt
% and then adds the heat transferred directly from the gases

clc
clear all
close all
load 'pressure_100%LTC'

% Cylinder and piston parameters
    compressR = 19;
    bore = 86;    % mm
    stroke = 86;  % mm
    crankr = 43;  % mm
    conrod = 160; % mm
    pistonmass = 1.1195;  % Piston mass (kg), includes pin mass (250g)
                          % and connecting rod mass fraction (271g)
    cylarea = 0.01*pi*bore^2/4; % cm2
    cylsurface = 1e-6*pi*bore*stroke; % m2
    pistonarea = pi*(bore-0.818)^2/4; % mm2
    displvol = 0.1*stroke*cylarea; % cm3
    clearvol = displvol/(compressR-1); % cm3
    clearheight = 0.01; % mm
    conrodcrankr = conrod/crankr;
    crankrconrod = crankr/conrod;

% Crank angle step
    stepCA = 0.1; % degrees
    istroke = 180/stepCA+1;
    irev = 360/stepCA+1;
    icycle = 720/stepCA+1;
    degCA = linspace(0,720,icycle)';
    radCA = pi*degCA/180;
    enginesp = 4000; % rev/min
    meansp = (0.002/60)*stroke*enginesp; % Mean piston velocity (m/s)
    borestep = 0.1; % mm
    boreheight = linspace(0,160,160/borestep+1); % mm

% Rings and skirt data
    Bcomprings = 0.002; % m
    Bocring = 0.00018; % m
    Lrings = pi*bore/1000; % m
    Licomprings = pi*78.8/1000; % m
    Lskirt = 0.04; % m
    Bskirt = 0.042; % m

    Acomprings = Bcomprings*Licomprings; % m2
    Askirt = Bskirt*Lskirt; % m2

    Ft1st = 17.9; % N
```

```
    Ft2nd = 24.1; % N
    Ftoc = 36.5;  % N

    factorFrings = 1.938; % Cameron's Principles of Lubrication pp 115-120
    factorFskirt = 2.86;
    factor6Wcrings = 0.1353;
    factor6Wocring = 0.1447;
    factor6Wskirt = 0.05979;

    temp1string = 180; % °C
    temp2ndring = 165; % °C
    tempocring = 140; % °C
    temptopskirt = 200; % °C
    tempmidskirt = 165; % °C
    tempbotskirt = 140; % °C
    tempdistskirt = interp1([27.3,50.7,74.1],[temptopskirt,tempmidskirt,...
                             tempbotskirt],[27.3 28:74 74.1],'spline');

% Oil data (SAE 5W 40)
    viscosityT1 = 95.3; % mm2/s (centistokes) at 40°C
    viscosityT2 = 14.4; % mm2/s (centistokes) at 100°C
    temp1 = 40 + 273.15; % K
    temp2 = 100 + 273.15; % K
    oildensity = 0.00085; % kg/cm3 at 15°C
    oilconductK = 0.138; % W/m.K

% Calculates time (s), height (mm), speed (m/s) and acceleration (m/s2)
for i=1:length(radCA),

    factor1 = sin(radCA(i));
    factor2 = cos(radCA(i));
    factor2a = cos(2*radCA(i));
    factor3 = conrodcrankr^2-sin(radCA(i))^2;
    factor4 = factor1/sqrt(conrodcrankr^2-factor1^2);

    time(i) = 30*radCA(i)/(pi*enginesp);

    instheight(i) = clearheight+crankr*(conrodcrankr+1-factor2-...
                                        sqrt(factor3));

    if abs(instheight(i)) < 1e-10
        instheight(i) = 0;
    end

    instsp(i) = 0.5*pi*meansp*factor1*(1+factor2/sqrt(factor3));

    if abs(instsp(i)) < 1e-10
        instsp(i) = 0;
    end

    exactac(i) = pi^2*enginesp*meansp*(factor2+(conrodcrankr^2*factor2a+...
                                       factor1^4)/factor3^(3/2))/60;

% Sets valves events
    if degCA(i) == 30 % Exhaust valves fully closed
        evc = i;
    elseif degCA(i) == 240 % Intake valves fully closed
        ivc = i;
    elseif degCA(i) == 480 % Exhaust valves starts to open
        evo = i;
    elseif degCA(i) == 700 % Intake valves starts to open
        ivo = i;
    end
end

% Calculates rings and skirt instantaneous height (mm)
instheight = instheight';
height1string = instheight + 9.7;
height2ndring = instheight + 18.3;
heightocring = instheight + 24.3;
```

```
heighttopskirt = instheight + 27.3;
heightmidskirt = instheight + 50.7;
heightbotskirt = instheight + 74.1;

% Calculates oil viscisty (kg/m.s) for the mean rings/skirt-bore
% temperatures
run viscosity_4000rpm_100LTC

% Calculates gas pressure acting on compression rings
% Filters measured cylinder gas pressure (bar)
a=14;
b=ones(1,a);
gaspressure = filtfilt(b,a,pressure_100LTC(:,7));

l=1;
for i=1:10*stepCA:length(gaspressure),
    cylpress(l,1) = gaspressure(i);
    l=l+1;
end
cylpress(icycle) = cylpress(1);

for i=1:length(radCA)-1,
    onecycle = 0;
    if i <= evc | i >= evo
        onecycle = icycle;
    end

    if i > ivc & i <= evo
        if i < irev
            interringpress(i) = cylpress(ivc)+(cylpress(i)-...
                                              cylpress(ivc))/3;
        else
            interringpress(i) = 1.25*cylpress(evo-1)+(cylpress(i)-...
                                                      cylpress(evo-1))/3;
        end
    elseif (i+onecycle > evo & i <= evc) | (i > evo & i <= evc+onecycle)
        if i <= evc
            cycle = degCA(icycle);
        else
            cycle = degCA(1);
        end
        interringpress(i,1) = 1.25*interp1([degCA(evo:icycle-1);...
                              degCA(1:evc)+degCA(icycle)],...
                             [cylpress(evo:icycle-1); cylpress(1:evc)],...
                              degCA(i)+cycle,'linear');
    else
        interringpress(i,1) = 0.75*cylpress(i);
    end
end
interringpress(icycle) = interringpress(1);

[y,x1] = max(interringpress);
for i=1:length(radCA),
    if cylpress(i) >= y & cylpress(i+1) <= y
        y2 = cylpress(i);
        x2 = i;
        y3 = cylpress(i+1);
        x3 = i+1;
        xshift = round(x2+(y2-y)*(x3-x2)/(y2-y3)-x1);
    end
end
interringpress = circshift(interringpress,xshift);
press1string = cylpress;
press2ndring = interringpress;

% Determination of forces (N)
sidethrustFt = [];
for i=1:length(radCA),

    factor1 = sin(radCA(i));
```

```
    factor4 = factor1/sqrt(conrodcrankr^2-factor1^2);

    pressureF(i) = 0.1*pistonarea*cylpress(i);

    inertiaF(i) = pistonmass*exactac(i);

    reactionNF(i,1) = factor4*(pressureF(i) - inertiaF(i));

% Rings load (N)
    W1string(i,1) = 1e5*Acomprings*press1string(i) + Ft1st;
    W2ndring(i,1) = 1e5*Acomprings*press2ndring(i) + Ft2nd;
    Wocring(i,1) = Ftoc/2;

% Rings friction coefficient
    coeff1string(i,1) = factorFrings*sqrt(viscosity1stringAF(i)*...
                                    abs(instsp(i))*Lrings/W1string(i));
    coeff2ndring(i,1) = factorFrings*sqrt(viscosity2ndringAF(i)*...
                                    abs(instsp(i))*Lrings/W2ndring(i));
    coeffocring(i,1) = factorFrings*sqrt(viscosityocringAF(i)*...
                                    abs(instsp(i))*Lrings/Wocring(i));

% Rings friction force (N)
    frictionF1string(i,1) = coeff1string(i)*W1string(i);
    frictionF2ndring(i,1) = coeff2ndring(i)*W2ndring(i);
    frictionFocring(i,1) = 2*coeffocring(i)*Wocring(i);

    if degCA(i) <= 180 | (degCA(i) > 360) & degCA(i) <= 540
        frictionF1string(i) = -frictionF1string(i);
        frictionF2ndring(i) = -frictionF2ndring(i);
        frictionFocring(i) = -frictionFocring(i);
    end
    ringsfrictionF(i,1) = frictionF1string(i) + frictionF2ndring(i) +...
                          frictionFocring(i);

% Set oil viscosity (kg/m.s) between skirt and thrust/non-thrust bore side
    if reactionNF(i) >= 0
        viscosityskirt = viscosityskirtA(i);
    else
        viscosityskirt = viscosityskirtF(i);
    end

% Friction quadratic equation coefficients
    C1 = 2*(inertiaF(i) - pressureF(i) + ringsfrictionF(i))*factor4 - ...
            viscosityskirt*abs(instsp(i))*Lskirt*(factorFskirt*factor4)^2;

    C2 = ((inertiaF(i) - pressureF(i) + ringsfrictionF(i))*factor4)^2;

% Solution of side thrust (N) equation
    radical(i,1) = C1^2-4*C2;

    if radical(i) < 0
        sidethrustFt(i,1) = reactionNF(i);
    else
        sidethrustFt(i,1) = (-C1 + sqrt(radical(i)))/2;
    end
    difference(i,1) = (sidethrustFt(i)-reactionNF(i))*100/sidethrustFt(i);

% Skirt friction coefficient
    coeffskirt(i,1) = factorFskirt*sqrt(abs(viscosityskirt*...
                      abs(instsp(i))*Lskirt/sidethrustFt(i)));

% Skirt friction force (N)
    if sidethrustFt(i,1) >= 0
        frictFskirtA(i,1) = factorFskirt*sqrt(viscosityskirt*...
                            abs(instsp(i))*Lskirt*sidethrustFt(i));
        frictFskirtF(i,1) = 0;
    else
        frictFskirtF(i,1) = -coeffskirt(i,1)*sidethrustFt(i); % N
        frictFskirtA(i,1) = 0;
    end
```

```
    if degCA(i) < 180 | (degCA(i) > 360) & degCA(i) < 540
        frictFskirtA(i) = -frictFskirtA(i);
        frictFskirtF(i) = -frictFskirtF(i);
    end
% Oil film thickness (m)
    oft1string(i,1) = Bcomprings*sqrt(factor6Wcrings*viscosity1stringAF(i)*...
                                 abs(instsp(i))*Lrings/W1string(i));
    oft2ndring(i,1) = Bcomprings*sqrt(factor6Wcrings*viscosity2ndringAF(i)*...
                                 abs(instsp(i))*Lrings/W2ndring(i));
    oftocring(i,1)  = Bocring*sqrt(factor6Wocring*viscosityocringAF(i)*...
                               abs(instsp(i))*Lrings/Wocring(i));

    if sidethrustFt(i) > 0
        oftskirtA(i,1) = Bskirt*sqrt(factor6Wskirt*viscosityskirt*...
                                 abs(instsp(i)*Lskirt/sidethrustFt(i)));
        oftskirtF(i,1) = 0;

        if oftskirtA(i) > 30*1e-6
            oftskirtA(i) = 30*1e-6;
        end
    elseif sidethrustFt(i) < 0
        oftskirtF(i,1) = Bskirt*sqrt(factor6Wskirt*viscosityskirt*...
                                 abs(instsp(i)*Lskirt/sidethrustFt(i)));
        oftskirtA(i,1) = 0;

        if oftskirtF(i) > 30*1e-6
            oftskirtF(i) = 30*1e-6;
        end
    else
        oftskirtA(i,1) = Bskirt*sqrt(factor6Wskirt*viscosityskirt*...
                                 abs(instsp(i)*Lskirt/sidethrustFt(i)));

        oftskirtF(i,1) = Bskirt*sqrt(factor6Wskirt*viscosityskirt*...
                                 abs(instsp(i)*Lskirt/sidethrustFt(i)));

        if oftskirtA(i) > 30*1e-6 | oftskirtF(i) > 30*1e-6
            oftskirtA(i) = 30*1e-6;
            oftskirtF(i) = 30*1e-6;
        end
    end

% Friction heat generation (kW)
% Rings
    frictQ1string(i,1) = 1e-3*abs(frictionF1string(i)*instsp(i));
    frictQ2ndring(i,1) = 1e-3*abs(frictionF2ndring(i)*instsp(i));
    frictQocring(i,1)  = 1e-3*abs(frictionFocring(i)*instsp(i));
% Skirt
    if sidethrustFt(i) > 0
        frictQskirtA(i,1) = 1e-3*abs(frictFskirtA(i)*instsp(i));
        frictQskirtF(i,1) = 0;
    elseif sidethrustFt(i) < 0
        frictQskirtF(i,1) = 1e-3*abs(frictFskirtF(i)*instsp(i));
        frictQskirtA(i,1) = 0;
    else
        frictQskirtA(i,1) = 1e-3*abs(frictFskirtA(i)*instsp(i));
        frictQskirtF(i,1) = 1e-3*abs(frictFskirtF(i)*instsp(i));
    end
end

l = 1;
for i=3:length(radCA),
    if sign(sidethrustFt(i)) ~= sign(sidethrustFt(i-1)) | degCA(i) == 720
        is(l) = i;
        l = l + 1;
    end
end

% Averaged conduction heat flux through the oil film (kW/m2) and
% frictional heat flux (kW/m2) over each crank angle step
```

```
for i=1:icycle,
    if i==1
        iadd = icycle-1;
    else
        iadd = 0;
    end
    distance(i,1) = 1e-3*abs(instheight(i)-instheight(i+iadd-1)); % m

% Rings
    if 0.5*(oft1string(i)+oft1string(i+iadd-1)) > 1.3e-7

        avgcondQ1stringA(i) = 1e-3*oilconductK*Bcomprings*...
                              (temp1string-0.5*(tempboreA1string(i)+...
                               tempboreA1string(i+iadd-1)))/(0.5*...
                              (oft1string(i)+oft1string(i+iadd-1))*...
                              (Bcomprings+distance(i)));

        avgcondQ1stringF(i) = 1e-3*oilconductK*Bcomprings*...
                              (temp1string-0.5*(tempboreF1string(i)+...
                               tempboreF1string(i+iadd-1)))/(0.5*...
                              (oft1string(i)+oft1string(i+iadd-1))*...
                              (Bcomprings+distance(i)));
    else
        avgcondQ1stringA(i) = 0;
        avgcondQ1stringF(i) = 0;
    end

    if 0.5*(oft2ndring(i)+oft2ndring(i+iadd-1)) > 1.3e-7

        avgcondQ2ndringA(i) = 1e-3*oilconductK*Bcomprings*...
                              (temp2ndring-0.5*(tempboreA2ndring(i)+...
                               tempboreA2ndring(i+iadd-1)))/(0.5*...
                              (oft2ndring(i)+oft2ndring(i+iadd-1))*...
                              (Bcomprings+distance(i)));

        avgcondQ2ndringF(i) = 1e-3*oilconductK*Bcomprings*...
                              (temp2ndring-0.5*(tempboreF2ndring(i)+...
                               tempboreF2ndring(i+iadd-1)))/(0.5*...
                              (oft2ndring(i)+oft2ndring(i+iadd-1))*...
                              (Bcomprings+distance(i)));
    else
        avgcondQ2ndringA(i) = 0;
        avgcondQ2ndringF(i) = 0;
    end

    if 0.5*(oftocring(i)+oftocring(i+iadd-1)) > 1.3e-7

        avgcondQocringA(i) = 1e-3*oilconductK*2*Bocring*...
                             (tempocring-0.5*(tempboreAocring(i)+...
                              tempboreAocring(i+iadd-1)))/(0.5*...
                             (oftocring(i)+oftocring(i+iadd-1))*...
                             (2*Bocring+distance(i)));

        avgcondQocringF(i) = 1e-3*oilconductK*2*Bocring*(tempocring-0.5*...
                             (tempboreFocring(i)+...
                              tempboreFocring(i+iadd-1)))/(0.5*...
                             (oftocring(i)+oftocring(i+iadd-1))*...
                             (2*Bocring+distance(i)));
    else
        avgcondQocringA(i) = 0;
        avgcondQocringF(i) = 0;
    end
    avgfrictQ1string(i) = 0.5*(frictQ1string(i)+frictQ1string(i+iadd-1))/...
                          (Lrings*(Bcomprings+distance(i)));

    avgfrictQ2ndring(i) = 0.5*(frictQ2ndring(i)+frictQ2ndring(i+iadd-1))/...
                          (Lrings*(Bcomprings+distance(i)));

    avgfrictQocring(i)  = 0.5*(frictQocring(i)+frictQocring(i+iadd-1))/...
                          (Lrings*(2*Bocring+distance(i)));
```

```
% Skirt
    if i==2
        avgoftA = 0.5*oftskirtA(i);
        avgoftF = 0.5*oftskirtF(i);
    else
        avgoftA = 0.5*(oftskirtA(i)+oftskirtA(i+iadd-1));
        avgoftF = 0.5*(oftskirtF(i)+oftskirtF(i+iadd-1));
    end

    if sidethrustFt(i) > 0

        avgcondQskirtA(i,1) = 1e-3*oilconductK*Bskirt*(tempmidskirt-0.5*...
                                  (tempboreAmidskirt(i)+...
                                   tempboreAmidskirt(i+iadd-1)))/...
                                  (avgoftA*(Bskirt+distance(i)));
        avgcondQskirtF(i,1) = 0;

    elseif sidethrustFt(i) < 0

        avgcondQskirtF(i,1) = 1e-3*oilconductK*Bskirt*(tempmidskirt-0.5*...
                                  (tempboreFmidskirt(i)+...
                                   tempboreFmidskirt(i+iadd-1)))/...
                                  (avgoftF*(Bskirt+distance(i)));
        avgcondQskirtA(i,1) = 0;

    else
        avgcondQskirtA(i,1) = 1e-3*oilconductK*Bskirt*...
                                  (tempmidskirt-0.5*(tempboreAmidskirt(i)+...
                                   tempboreAmidskirt(i+iadd-1)))/...
                                  (avgoftA*(Bskirt+distance(i)));

        avgcondQskirtF(i,1) = 1e-3*oilconductK*Bskirt*(tempmidskirt-0.5*...
                                  (tempboreFmidskirt(i)+...
                                   tempboreFmidskirt(i+iadd-1)))/...
                                  (avgoftF*(Bskirt+distance(i)));
    end

    avgfrictQskirtA(i,1) = 0.5*(frictQskirtA(i)+frictQskirtA(i+iadd-1))/...
                               (Lskirt*(Bskirt+distance(i)));
    avgfrictQskirtF(i,1) = 0.5*(frictQskirtF(i)+frictQskirtF(i+iadd-1))/...
                               (Lskirt*(Bskirt+distance(i)));
    oftmeanA(i) = avgoftA;
    oftmeanF(i) = avgoftF;
end

% Cycle conduction heat flux (kW/m2)
l = 2;
iadd = 0;
for i=2:istroke,
% Rings
% Thrust side
    cyclecondQ1stringA(l,1) = stepCA*(avgcondQ1stringA(i)+...
                                      avgcondQ1stringA(irev+2-i)+...
                                      avgcondQ1stringA(irev-1+i)+...
                                      avgcondQ1stringA(icycle+2-i))/720;

    cyclecondQ2ndringA(l,1) = stepCA*(avgcondQ2ndringA(i)+...
                                      avgcondQ2ndringA(irev+2-i)+...
                                      avgcondQ2ndringA(irev-1+i)+...
                                      avgcondQ2ndringA(icycle+2-i))/720;

    cyclecondQocringA(l,1) = stepCA*(avgcondQocringA(i)+...
                                     avgcondQocringA(irev+2-i)+...
                                     avgcondQocringA(irev-1+i)+...
                                     avgcondQocringA(icycle+2-i))/720;

% Skirt
% Thrust side
    cyclecondQskirtA(l,1) = stepCA*(avgcondQskirtA(i)+...
```

```
                                    avgcondQskirtA(irev+2-i)+...
                                    avgcondQskirtA(irev-1+i)+...
                                    avgcondQskirtA(icycle+2-i))/720;

% Cycle friction heat flux (kW/m2)
% Rings
    cyclefrictQ1string(l,1) = stepCA*(avgfrictQ1string(i)+...
                                      avgfrictQ1string(irev+2-i)+...
                                      avgfrictQ1string(irev-1+i)+...
                                      avgfrictQ1string(icycle+2-i))/720;

    cyclefrictQ2ndring(l,1) = stepCA*(avgfrictQ2ndring(i)+...
                                      avgfrictQ2ndring(irev+2-i)+...
                                      avgfrictQ2ndring(irev-1+i)+...
                                      avgfrictQ2ndring(icycle+2-i))/720;

    cyclefrictQocring(l,1) = stepCA*(avgfrictQocring(i)+...
                                     avgfrictQocring(irev+2-i)+...
                                     avgfrictQocring(irev-1+i)+...
                                     avgfrictQocring(icycle+2-i))/720;

% Skirt
% Thrust side
    cyclefrictQskirtA(l,1) = stepCA*(avgfrictQskirtA(i)+...
                                     avgfrictQskirtA(irev+2-i)+...
                                     avgfrictQskirtA(irev-1+i)+...
                                     avgfrictQskirtA(icycle+2-i))/720;
    l=l+1;
end

cyclecondQ1stringA(1) = cyclecondQ1stringA(2);
cyclecondQ2ndringA(1) = cyclecondQ2ndringA(2);
cyclecondQocringA(1) = cyclecondQocringA(2);

cyclefrictQ1string(1) = cyclefrictQ1string(2);
cyclefrictQ2ndring(1) = cyclefrictQ2ndring(2);
cyclefrictQocring(1) = cyclefrictQocring(2);

cyclecondQskirtA(1) = cyclecondQskirtA(2);
cyclefrictQskirtA(1) = cyclefrictQskirtA(2);

% Integration of heat flux at bore locations
% First ring
intcondQ1stringA = zeros(1,length(boreheight));
intfrictQ1string = zeros(1,length(boreheight));

for i=1:istroke,
    if i==1
        kini = 1+int16(height1string(i)/borestep);
    else
        kini = 1+int16(height1string(i-1)/borestep);
    end
    kfin = int16(height1string(i)/borestep)+2/borestep;

    for k=kini:kfin,
        intcondQ1stringA(k) = intcondQ1stringA(k) + cyclecondQ1stringA(i);
        intfrictQ1string(k) = intfrictQ1string(k) + cyclefrictQ1string(i);
    end
end
intcondandfrictQ1stringA = intcondQ1stringA+intfrictQ1string;

% Second ring
intcondQ2ndringA = zeros(1,length(boreheight));
intfrictQ2ndring = zeros(1,length(boreheight));

for i=1:istroke,
    if i==1
        kini = 1+int16(height2ndring(i)/borestep);
    else
        kini = 1+int16(height2ndring(i-1)/borestep);
```

```
    end
    kfin = int16(height2ndring(i)/borestep)+2/borestep;

    for k=kini:kfin,
        intcondQ2ndringA(k) = intcondQ2ndringA(k) + cyclecondQ2ndringA(i);
        intfrictQ2ndring(k) = intfrictQ2ndring(k) + cyclefrictQ2ndring(i);
    end
end

intcondandfrictQ2ndringA = intcondQ2ndringA+intfrictQ2ndring;

% Oil control ring
intcondQocringA = zeros(1,length(boreheight));
intfrictQocring = zeros(1,length(boreheight));

for i=1:istroke,
    if i==1
        kini = 1+int16(heightocring(i)/borestep);
    else
        kini = 1+int16(heightocring(i-1)/borestep);
    end
    kfin = int16(heightocring(i)/borestep)+2/borestep;

    for k=kini:kfin,
        intcondQocringA(k) = intcondQocringA(k) + cyclecondQocringA(i);
        intfrictQocring(k) = intfrictQocring(k) + cyclefrictQocring(i);
    end
end

intcondandfrictQocringA = intcondQocringA+intfrictQocring;

% Integration of skirt heat transfer at bore locations
intcondQskirtA = zeros(1,length(boreheight));
intfrictQskirtA = zeros(1,length(boreheight));

for k=1:length(boreheight),
    for i=2:istroke,
        if heighttopskirt(i) <= k*borestep & heightbotskirt(i) > k*borestep
            intcondQskirtA(k) = intcondQskirtA(k) + cyclecondQskirtA(i);
            intfrictQskirtA(k) = intfrictQskirtA(k) + cyclefrictQskirtA(i);
        end
    end
end
intcondandfrictQskirtA = intcondQskirtA + intfrictQskirtA;

k = 0;
for i=1:length(boreheight),
    if boreheight(i) > 9.9 & k == 0
        i1st1 = i;
        k = 1;
        intcondQ1stringA(1:i1st1-1) = 0;
    end

    if boreheight(i) > 18.3 & k == 1
        i2nd1 = i-1;
        k = 2;
        intcondQ2ndringA(1:i2nd1-1) = 0;
    end

    if boreheight(i) > 24.3 & k == 2
        ioc1 = i-1;
        k = 3;
        intcondQocringA(1:ioc1-1) = 0;
    end

    if boreheight(i) > 27.3 & k == 3
        itopskirt = i-1;
        k = 4;
        intcondQskirtA(1:itopskirt-1) = 0;
        intfrictQskirtA(1:itopskirt-1) = 0;
```

```
    end

    if boreheight(i) > 95.7 & k == 4
        i1st2 = i-1;
        k = 5;
        intcondQ1stringA(i1st2:length(intcondQ1stringA)) = 0;
    end

    if boreheight(i) > 100 & k == 5
        iend = i-1;
        k = 6;
        intcondQ2ndringA(iend+1:length(intcondQ2ndringA)) = 0;
        intcondQocringA(iend+1:length(intcondQocringA)) = 0;
        intcondQskirtA(iend+1:length(intcondQskirtA)) = 0;
        intfrictQskirtA(iend+1:length(intfrictQskirtA)) = 0;
    end
end
intcondQringsA = intcondQ1stringA + intcondQ2ndringA + intcondQocringA;

% Smooths friction heat flux
a=4;
b=ones(1,a);
intfrictQ1string = filtfilt(b,a,intfrictQ1string);
intfrictQ2ndring = filtfilt(b,a,intfrictQ2ndring);
intfrictQocring = filtfilt(b,a,intfrictQocring);

intfrictQrings = intfrictQ1string + intfrictQ2ndring + intfrictQocring;
intcondandfrictQringsA = intcondQringsA + intfrictQrings;

% Finds the piston height at relevant crank angle
for i = 1:istroke,
    if degCA(i) == 60,
        h60 = instheight(i);
    elseif degCA(i) == 120,
        h120 = instheight(i);
    elseif degCA(i) == 180,
        h180 = instheight(i);
    else
        continue
    end
end

k = 1;
for i=1:length(boreheight),
    if  boreheight(i) >= h60 & k == 1;
        i60 = i-1;
        k = 2;
    elseif boreheight(i) >= h120 & k == 2;
        i120 = i-1;
        k = 3;
    elseif boreheight(i) >= h180 & k == 3;
        i180 = i-1;
        k = 4;
    else
        continue
    end
end

% Calculates convective heat flux at location A
run convective_heatflux_thrust

% Calculates experimental heat flux
run experimental_heatflux_thrust

% Plots predicted heat fluxes vs experimental heat flux
h=figure;
set(h,'PaperOrientation','landscape',...
      'Name','Total heat flux and experimental heat flux (thrust side)',...
      'PaperOrientation','landscape',...
      'PaperPosition',[0.6345 1.129 28.41 18.73],'PaperSize',[29.68 20.98])
```

```
hold on
plot(boreheight(i1st1:iend),intcondQringsA(i1st1:iend),'b-',...
     boreheight(i1st1:iend),intcondandfrictQringsA(i1st1:iend),'g-',...
     boreheight(i1st1:iend),intcondandfrictQringsA(i1st1:iend)+...
                            intcondQskirtA(i1st1:iend),'m-',...
     boreheight(i1st1:iend),intcondandfrictQringsA(i1st1:iend)+...
                            intcondandfrictQskirtA(i1st1:iend),'c-',...
     boreheight(i1st1:iend),intcondandfrictQringsA(i1st1:iend)+...
                            intcondandfrictQskirtA(i1st1:iend)+...
     heatfluxCbore(i1st1:iend),'k--','LineWidth',1.5)
plot(boreheight(i1st1:iend),experheatA(i1st1:iend),'r--','LineWidth',1.5)
plot(location,measvaluesA,'rs','MarkerSize',7,'MarkerEdgeColor','r',...
                               'MarkerFaceColor','r')
plot(boreheight(i180),0:0.5:500,'k')
set(gca,'XLim',[0 110],'XTick',[0:10:110],'Fontsize',14)
set(gca,'YLim',[0 500],'YTick',[0:50:500],'Fontsize',14)
ylabel('{Heat flux (kW/m}^{2})','FontSize',16)
xlabel('Distance from top deck (mm)','FontSize',16)
text(-0.7,512,'TDC','FontSize',12);
text(h180-3.4,512,'BDC','FontSize',12);
legh = legend;
legend(legh,...
      {'Conductive heat flux from the piston ring pack',...
       'Conductive + frictional heat fluxes from the piston ring pack',...
      ['Conductive + frictional heat fluxes from the piston ring pack + '...
       'conductive heat flux from the piston skirt'],...
      ['Conductive + frictional heat fluxes from the piston ring pack + '...
       'conductive + frictional heat fluxes from the piston skirt'],...
      ['Conductive + frictional heat fluxes from the piston ring pack + '...
       'conductive + frictional heat fluxes from the piston skirt + '...
       'heat flux from cylinder gases'],'Experimental heat flux'},...
       'FontSize',10,'location','North');
grid on
box on
```

Appendix E. Uncertainty Analysis Equations

E.1 Wall Surface Temperature

The method described by Holman (1994) gives the uncertainty w_R of a quantity R, which is a direct result derived from independent variables $x_1, x_2, x_3, \ldots, x_n$, as a function of the uncertainties $w_1, w_2, w_3, \ldots, w_n$ of the independent variables according to the equation:

$$w_R^2 = \left(\frac{\partial R}{\partial x_1} w_1\right)^2 + \left(\frac{\partial R}{\partial x_2} w_2\right)^2 + \left(\frac{\partial R}{\partial x_3} w_3\right)^2 + \ldots + \left(\frac{\partial R}{\partial x_n} w_n\right)^2 \tag{E.1}$$

In this case, R represents the temperature on the gas-side surface of the cylinder wall T_{wg}, which is given by the equation of the temperature distribution in a cylindrical system where heat flows in the radial direction (Equation 4.4).

$$T_r = m \ln(r / r_B) + b \tag{E.2}$$

In Equation E.2, the temperature gradient m and the independent term b are obtained from three temperature measurements T_1, T_2, T_3 at the corresponding radii r_1, r_2, r_3 (Equations 4.5 and 4.6).

$$m = \frac{3(T_1 \ln r_1 + T_2 \ln r_2 + T_3 \ln r_3) - (\ln r_1 + \ln r_2 + \ln r_3)(T_1 + T_2 + T_3)}{3[(\ln r_1)^2 + (\ln r_2)^2 + (\ln r_3)^2] - (\ln r_1 + \ln r_2 + \ln r_3)^2} \tag{E.3}$$

$$b = \frac{(T_1 + T_2 + T_3)\{[\ln(r_1 / r_B)]^2 + [\ln(r_2 / r_B)]^2 + [\ln(r_3 / r_B)]^2\}}{3[(\ln r_1)^2 + (\ln r_2)^2 + (\ln r_3)^2] - (\ln r_1 + \ln r_2 + \ln r_3)^2} -$$
$$\frac{[T_1 \ln\{r_1 / r_B) + T_2 \ln(r_2 / r_B) + T_3 \ln(r_3 / r_B)][\ln(r_1 / r_B) + \ln(r_2 / r_B) + \ln(r_3 / r_B)]}{3[(\ln r_1)^2 + (\ln r_2)^2 + (\ln r_3)^2] - (\ln r_1 + \ln r_2 + \ln r_3)^2} \tag{E.4}$$

Equation E.2 shows that the temperature T_r at a radius r, equal to the radius of the cylinder bore r_B, is equal to the independent term b. Hence, the temperature T_{wg} can be written as:

$$T_{wg} = \frac{(T_1 + T_2 + T_3)\{[\ln(r_1 / r_B)]^2 + [\ln(r_2 / r_B)]^2 + [\ln(r_3 / r_B)]^2\}}{3[(\ln r_1)^2 + (\ln r_2)^2 + (\ln r_3)^2] - (\ln r_1 + \ln r_2 + \ln r_3)^2} -$$
$$\frac{[T_1 \ln\{r_1 / r_B) + T_2 \ln(r_2 / r_B) + T_3 \ln(r_3 / r_B)][\ln(r_1 / r_B) + \ln(r_2 / r_B) + \ln(r_3 / r_B)]}{3[(\ln r_1)^2 + (\ln r_2)^2 + (\ln r_3)^2] - (\ln r_1 + \ln r_2 + \ln r_3)^2} \tag{E.5}$$

Substitution of R and $x_1, x_2, x_3, \ldots, x_n$ in Equation E.1 by T_{wg} and the independent variables T_1, T_2, T_3, r_1, r_2, r_3, and r_B respectively, gives the equation for the uncertainty of the temperature on the gas-side surface of the cylinder wall $w_{T_{wg}}$.

$$w_{T_{wg}}^2 = \left[\left(\frac{\partial T_{wg}}{\partial T_1}\right)^2 + \left(\frac{\partial T_{wg}}{\partial T_2}\right)^2 + \left(\frac{\partial T_{wg}}{\partial T_3}\right)^2\right] w_T^2 + \left[\left(\frac{\partial T_{wg}}{\partial r_1}\right)^2 + \left(\frac{\partial T_{wg}}{\partial r_2}\right)^2 + \left(\frac{\partial T_{wg}}{\partial r_3}\right)^2\right] w_r^2 + \left(\frac{\partial T_{wg}}{\partial r_B} w_{r_B}\right)^2 \quad \text{(E.6)}$$

In Equation E.6, the partial derivatives of T_{wg} are obtained by differentiation of Equation E.5. The resulting equations for the partial derivatives are as follows:

$$\frac{\partial T_{wg}}{\partial T_1} = \frac{(\ln r_2)^2 + (\ln r_3)^2 - (\ln r_2 + \ln r_3)\ln r_1}{A} \quad \text{(E.7)}$$

$$\frac{\partial T_{wg}}{\partial T_2} = \frac{(\ln r_1)^2 + (\ln r_3)^2 - (\ln r_1 + \ln r_3)\ln r_2}{A} \quad \text{(E.8)}$$

$$\frac{\partial T_{wg}}{\partial T_3} = \frac{(\ln r_1)^2 + (\ln r_2)^2 - (\ln r_1 + \ln r_2)\ln r_3}{A} \quad \text{(E.9)}$$

$$\frac{\partial T_{wg}}{\partial r_1} = \frac{[T_1(2\ln r_B - \ln r_2 - \ln r_3) + T_2(2\ln r_1 - \ln r_2 - \ln r_B) + T_3(2\ln r_1 - \ln r_3 - \ln r_B)]}{r_1 A} - \frac{2B(2\ln r_1 - \ln r_2 - \ln r_3)}{r_1 A^2} \quad \text{(E.10)}$$

$$\frac{\partial T_{wg}}{\partial r_2} = \frac{[T_2(2\ln r_B - \ln r_1 - \ln r_3) + T_1(2\ln r_2 - \ln r_1 - \ln r_B) + T_3(2\ln r_2 - \ln r_3 - \ln r_B)]}{r_2 A} - \frac{2B(2\ln r_2 - \ln r_1 - \ln r_3)}{r_2 A^2} \quad \text{(E.11)}$$

$$\frac{\partial T_{wg}}{\partial r_3} = \frac{[T_3(2\ln r_B - \ln r_1 - \ln r_2) + T_1(2\ln r_3 - \ln r_1 - \ln r_B) + T_2(2\ln r_3 - \ln r_2 - \ln r_B)]}{r_3 A} - \frac{2B(2\ln r_3 - \ln r_1 - \ln r_2)}{r_3 A^2} \quad \text{(E.12)}$$

$$\frac{\partial T_{wg}}{\partial r_B} = \frac{T_1[\ln(r_2/r_B) + \ln(r_3/r_B) - 2\ln(r_1/r_B)] + T_2[\ln(r_1/r_B) + \ln(r_3/r_B) - 2\ln(r_2/r_B)]}{r_B A} + \frac{T_3[\ln(r_1/r_B) + \ln(r_2/r_B) - 2\ln(r_3/r_B)]}{r_B A} \quad \text{(E.13)}$$

where A and B are the denominator and numerator of Equation E.5 respectively.

E.2 Heat Flux

Fourier's law of heat conduction applied to a cylindrical system where heat flows in the radial direction gives the equation of the one-dimensional heat flux normal to a particular surface of radius r, $\dot{q}_r$ (Equation 4.8):

$$\dot{q}_r = -k\frac{m}{r} \quad \text{(E.14)}$$

In Equation E.14, the temperature gradient m can be replaced by Equation E.3. The resulting equation written for the gas-side surface of the cylinder wall, that is for a radius r_B, is as follows:

$$\dot{q}_{r_B} = -\frac{k}{r_B}\left[\frac{3(T_1 \ln r_1 + T_2 \ln r_2 + T_3 \ln r_3) - (\ln r_1 + \ln r_2 + \ln r_3)(T_1 + T_2 + T_3)}{3[(\ln r_1)^2 + (\ln r_2)^2 + (\ln r_3)^2] - (\ln r_1 + \ln r_2 + \ln r_3)^2}\right] \tag{E.15}$$

Substitution of R and x_1, x_2, x_3,..., x_n in Equation E.1 by $\dot{q}_{r_B}$ and the independent variables k, T_1, T_2, T_3, r_1, r_2, r_3, and r_B respectively, gives the equation for the uncertainty in the calculation of the radial heat flux normal to the gas-side surface of the cylinder wall $w_{\dot{q}_{rB}}$.

$$w_{\dot{q}_{rB}}^2 = \left[\left(\frac{\partial \dot{q}_{r_B}}{\partial T_1}\right)^2 + \left(\frac{\partial \dot{q}_{r_B}}{\partial T_2}\right)^2 + \left(\frac{\partial \dot{q}_{r_B}}{\partial T_3}\right)^2\right] w_T^2 + \left[\left(\frac{\partial \dot{q}_{r_B}}{\partial r_1}\right)^2 + \left(\frac{\partial \dot{q}_{r_B}}{\partial r_2}\right)^2 + \left(\frac{\partial \dot{q}_{r_B}}{\partial r_3}\right)^2\right] w_r^2 + \left(\frac{\partial \dot{q}_{r_B}}{\partial r_B} w_{r_B}\right)^2 + \left(\frac{\partial \dot{q}_{r_B}}{\partial k} w_k\right)^2 \tag{E.16}$$

The partial derivatives of $\dot{q}_{r_B}$ in Equation E.16 are found by differentiation of Equation E.15. The resulting equations are shown below.

$$\frac{\partial \dot{q}_{r_B}}{\partial T_1} = -\frac{k}{r_B}\frac{\partial m}{\partial T_1} = -\frac{k}{r_B}\left(\frac{2\ln r_1 - \ln r_2 - \ln r_3}{A}\right) \tag{E.17}$$

$$\frac{\partial \dot{q}_{r_B}}{\partial T_2} = -\frac{k}{r_B}\frac{\partial m}{\partial T_2} = -\frac{k}{r_B}\left(\frac{2\ln r_2 - \ln r_1 - \ln r_3}{A}\right) \tag{E.18}$$

$$\frac{\partial \dot{q}_{r_B}}{\partial T_3} = -\frac{k}{r_B}\frac{\partial m}{\partial T_3} = -\frac{k}{r_B}\left(\frac{2\ln r_3 - \ln r_1 - \ln r_2}{A}\right) \tag{E.19}$$

$$\frac{\partial \dot{q}_{r_B}}{\partial r_1} = -\frac{k}{r_B}\frac{\partial m}{\partial r_1} = -\frac{k}{r_B}\left[\frac{(2T_1 - T_2 - T_3)A - B(4\ln r_1 - 2\ln r_2 - 2\ln r_3)}{r_1 A^2}\right] \tag{E.20}$$

$$\frac{\partial \dot{q}_{r_B}}{\partial r_2} = -\frac{k}{r_B}\frac{\partial m}{\partial r_2} = -\frac{k}{r_B}\left[\frac{(2T_2 - T_1 - T_3)A - B(4\ln r_2 - 2\ln r_1 - 2\ln r_3)}{r_2 A^2}\right] \tag{E.21}$$

$$\frac{\partial \dot{q}_{r_B}}{\partial r_3} = -\frac{k}{r_B}\frac{\partial m}{\partial r_3} = -\frac{k}{r_B}\left[\frac{(2T_3 - T_1 - T_2)A - B(4\ln r_3 - 2\ln r_1 - 2\ln r_2)}{r_3 A^2}\right] \tag{E.22}$$

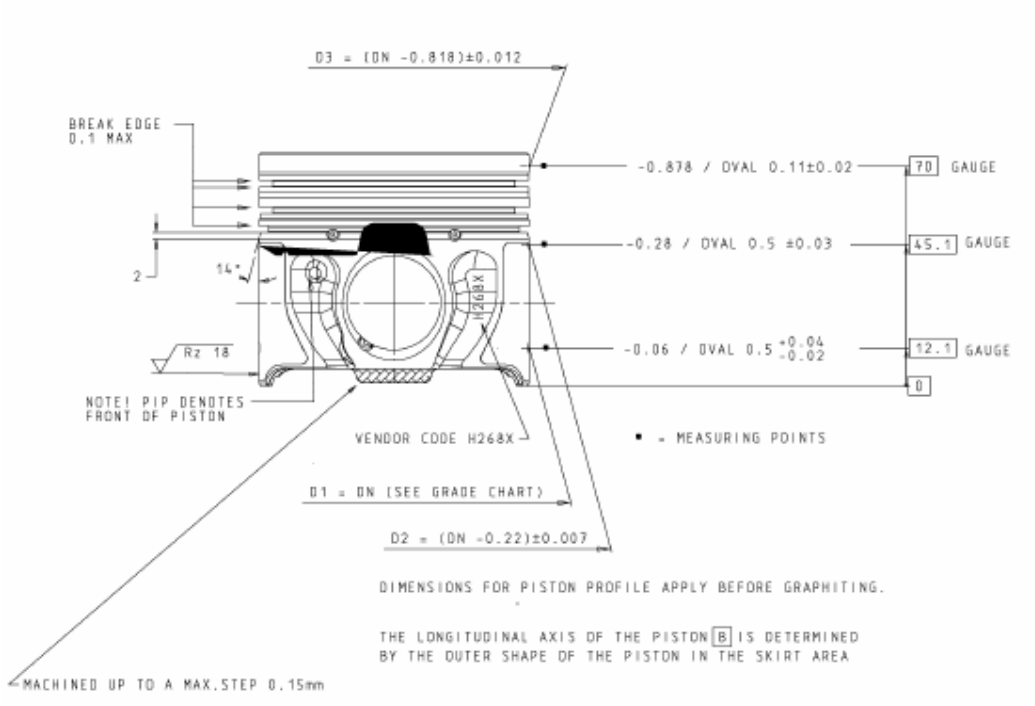
$$\frac{\partial \dot{q}_{r_B}}{\partial r_B} = \frac{k}{r_B{}^2}\left[\frac{3(T_1 \ln r_1 + T_2 \ln r_2 + T_3 \ln r_3) - (\ln r_1 + \ln r_2 + \ln r_3)(T_1 + T_2 + T_3)}{3[(\ln r_1)^2 + (\ln r_2)^2 + (\ln r_3)^2] - (\ln r_1 + \ln r_2 + \ln r_3)^2}\right] \tag{E.23}$$

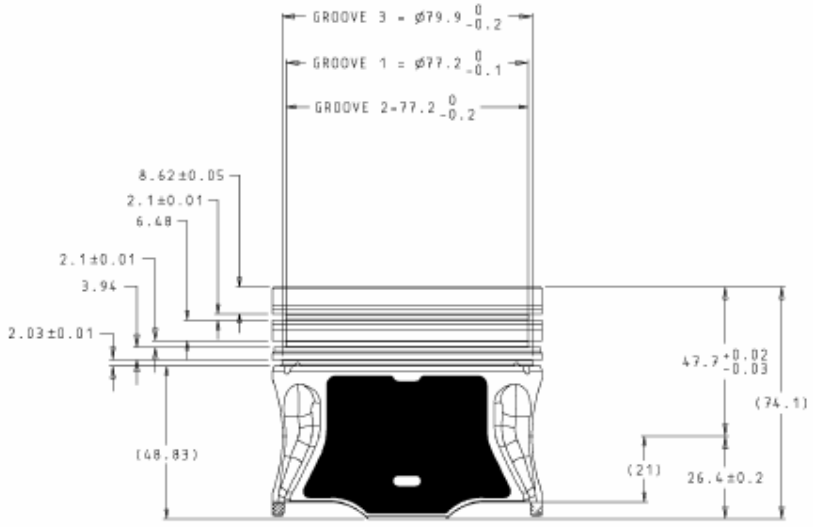
$$\frac{\partial \dot{q}_{r_B}}{\partial k} = -\frac{1}{r_B}\left[\frac{3(T_1 \ln r_1 + T_2 \ln r_2 + T_3 \ln r_3) - (\ln r_1 + \ln r_2 + \ln r_3)(T_1 + T_2 + T_3)}{3[(\ln r_1)^2 + (\ln r_2)^2 + (\ln r_3)^2] - (\ln r_1 + \ln r_2 + \ln r_3)^2}\right] \tag{E.24}$$

where A and B are the denominator and numerator of Equation E.3 respectively.

Appendix F. Piston and Lubricant Data

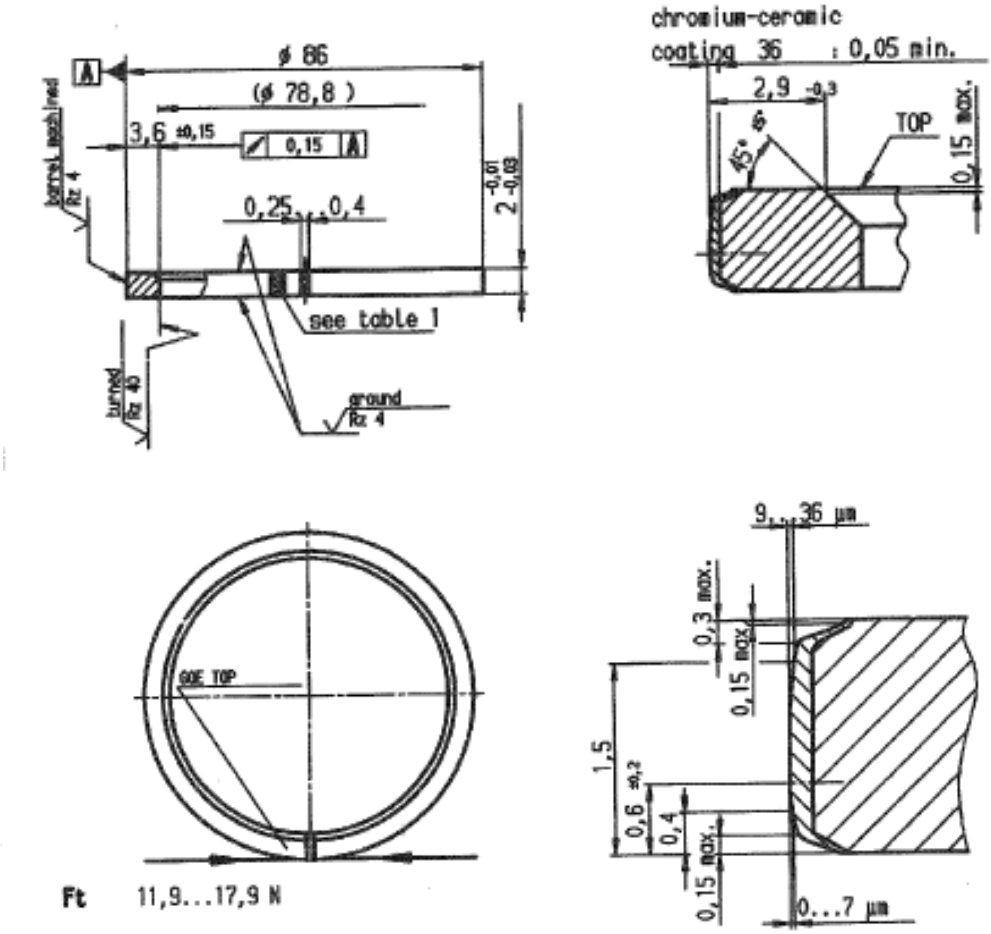
F.1 Piston Dimensions





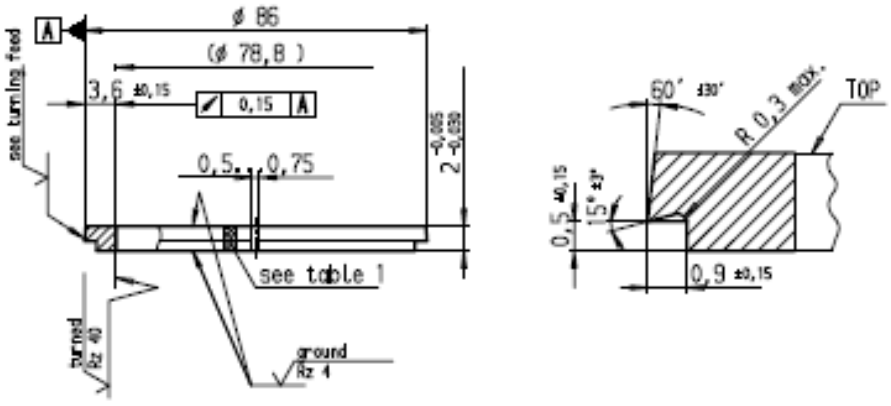
Adapted from data provided by Federal Mogul (Drawing 3S7Q-6110-DAA).

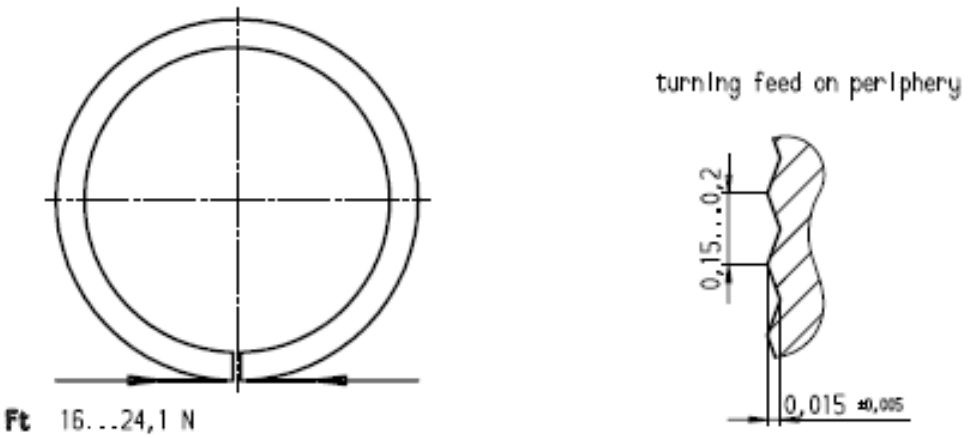
F.2 First Compresssion Ring



Adapted from data provided by Federal Mogul (Drawing No. 3S7Q-6150-BAA).

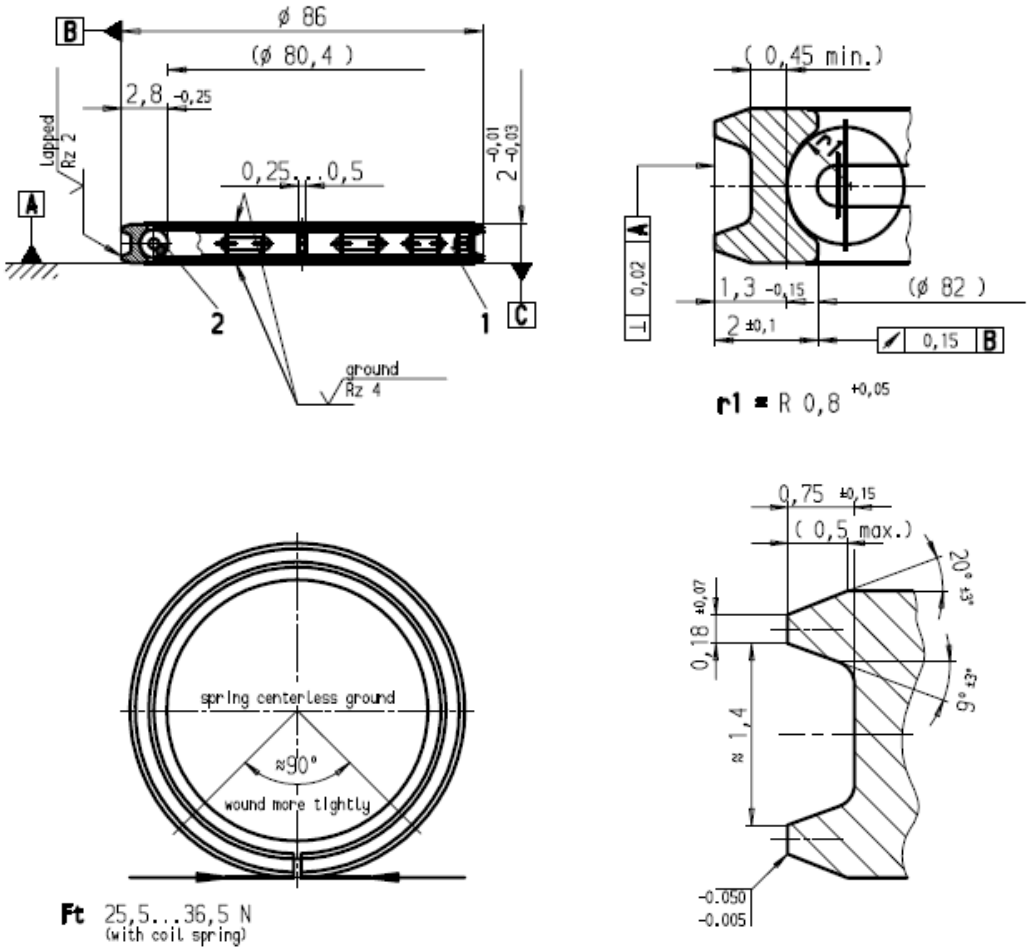
F.3 Second Compresssion Ring

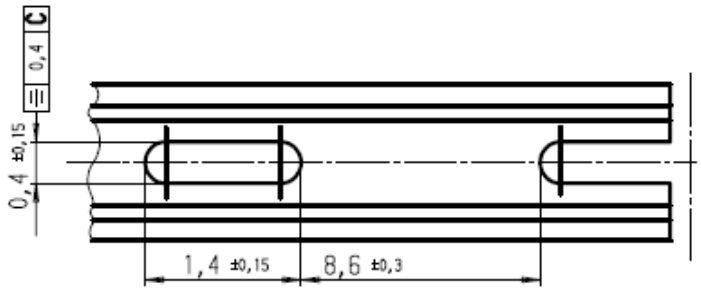




Adapted from data provided by Federal Mogul (Drawing No. K 00 685 155 0).

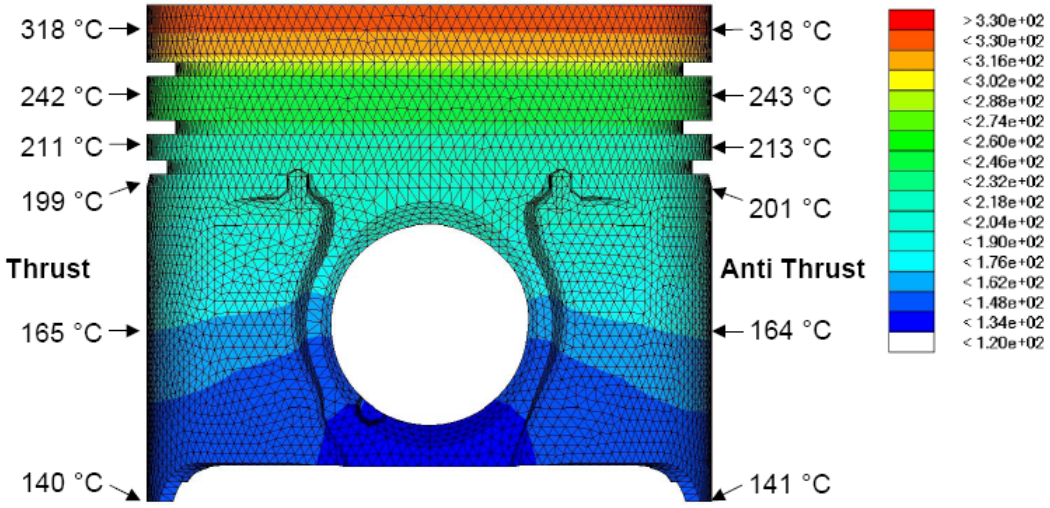
F.4 Oil Control Ring





Adapted from data provided by Federal Mogul (Drawing No. K 03 549 844 1).

F.5 Piston Temperature Distribution at Maximum Power

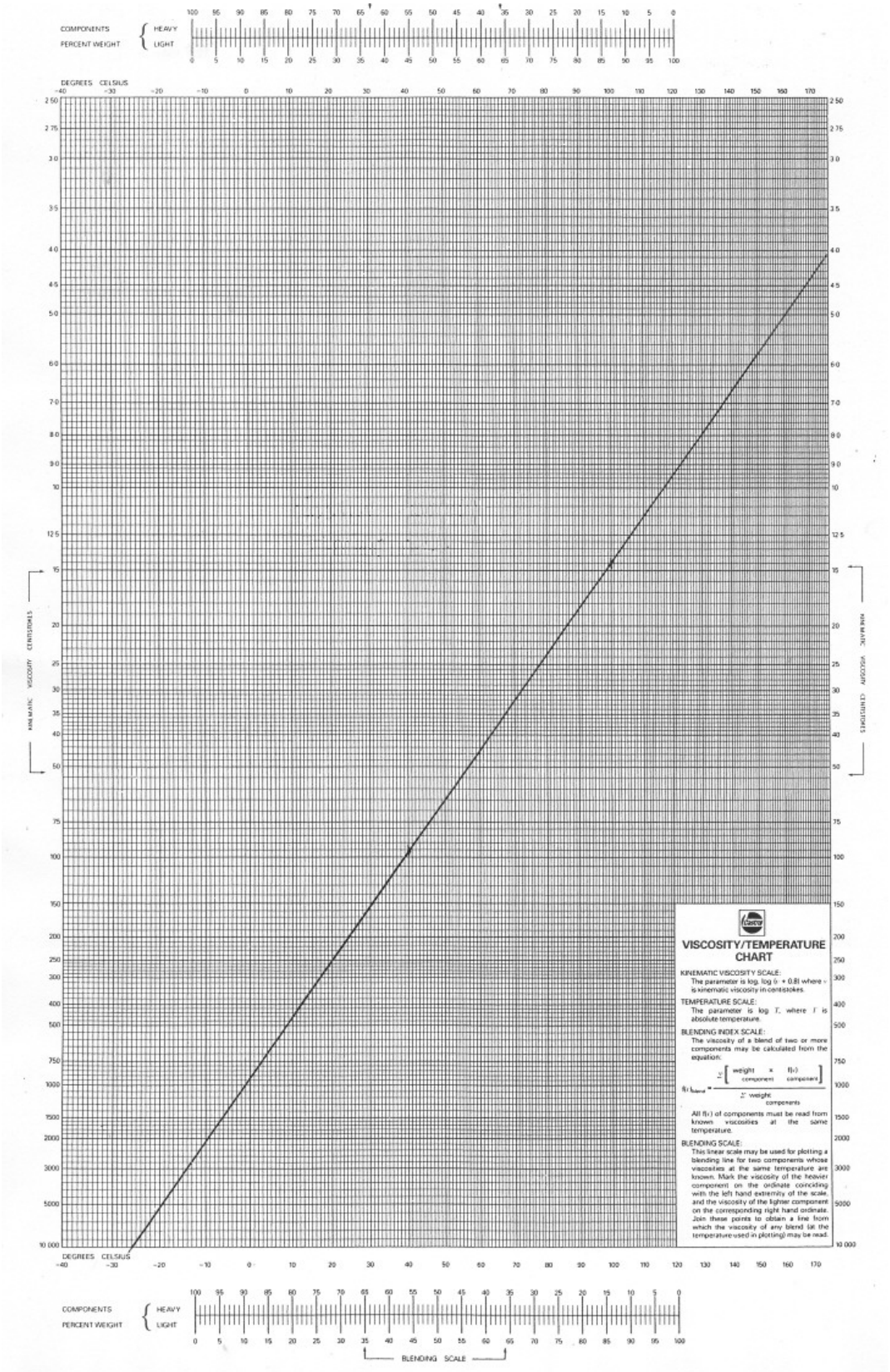


Adapted from data provided by the manufacturer (Federal Mogul, 2005).

F.6 Lubricant Oil

Synthetic lubricant Castrol 505 01 5W-40 formulated for use in advanced diesel engines. Specifications: API SJ/CF, Ford WSS M2C917-A, VW 505 01 and MB 229.1.

Property	Unit	Value
Relative Density at 15ºC	g/cm^3	0.85
Viscosity at 40ºC	mm^2/s	95.3
Viscosity at 100ºC	mm^2/s	14.4
Viscosity Index		156
CCS -30ºC	mPa.s	6000
Total Base Number	mgKOH/g	10.2
Flash Point (COC)	^0C	226
Pour Point	^0C	-51
Sulphated Ash	%m	1.2



Lubricant data provided by Castrol (UK) Ltd.